PONTIFICIA UNIVERSIDAD CATOLICA DEL ECUADOR

FACULTAD DE INGENIERÍA

ESCUELA DE SISTEMAS

ESTUDIO DE LAS REDES GPS-WI-FI Y SU APLICACIÓN EN LA MONITORIZACIÓN Y LOCALIZACIÓN DE DISPOSITIVOS MÓVILES

DIEGO PONCE

JUAN PRADO DÍAZ

Director:

Dr. Gustavo Chafla A.

Quito, enero del 2017

AGRADECIMIENTO

A mi Padre Juan, el hombre que más admiro en el mundo, por la entrega y sacrificio incondicional que has brindado a tu familia, por todo el conocimiento que me has dado, tu siempre has sido mi primer y el más importante de mis mentores y maestros, porque con amor y sabiduría me lograste poner en lo más alto, por enseñarme que sobre todo siempre estará la familia, porque nunca perdiste la fe en mí. "El gran hombre que soy ahora, se lo debo al gran hombre que tú siempre serás. Dios te pague Papito".

A mi madre Patricia, la mujer que siempre amaré y tendré su amor incondicional, gracias por la sabiduría y constancia que solo una madre como tú tiene para sacar adelante su hogar, porque desde que tengo memoria no descansaste un solo día para formar a tu familia con paciencia, cariño y sabiduría. "Gracias mamita por luchar por tu vida, porque solo así nos diste lo mejor de nuestras vidas".

A mi hermana Andrea por todo su apoyo incondicional, por sus ánimos en los momentos más tensos y oscuros, ayudándome a mantenerme en pie, y sabiendo que puedo contar contigo en todo momento, logrando sacarme de la negra soledad que un hijo único puede tener, gracias por convertirte en mi mejor amiga.

A mi compañero de disertación y querido amigo Diego, por toda su entrega y dedicación en este proyecto, por la fortaleza y sacrificio que supo brindar y sacarlo adelante, admirado siempre por tu conocimiento y la pasión que pones a todo lo que has creado, un muy afectuoso "Gracias Ponchito".

Al Dr. Gustavo Chafla por brindarnos la oportunidad de culminar nuestra vida universitaria y guiarnos durante este arduo camino hasta completar el objetivo deseado. De igual manera cordial agradecimiento a nuestros correctores, Fabián de la Cruz y Damián Nicolalde por toda su ayuda y colaboración en la elaboración del presente trabajo de disertación.

Un agradecimiento especial a mis queridos amigos Cristhian Peñafiel, Oscar Córdova, Christian Vega, Carlos Rosales, Steven Coronado y Johnny Arias con los cuales ha sido un inmenso placer haber compartido mis últimos años de universidad, por todo su apoyo e invaluable amistad, por haberme dado la comodidad de estar con ustedes como si fueran mi segunda familia, un inolvidable "Gracias Muchachos".

A todas las personas que hicieron de este sueño, una hermosa realidad, un cariñoso "Dios les pague".

Juan Prado Díaz

Existen personas que están destinadas a estar con nosotros y acompañarnos en este camino que llamamos vida, con el único propósito de hacernos mejores, enseñándonos de forma vivencial como debe comportarse un hombre, un hombre de verdad.

Y yo aprendí, que un hombre es humilde y respetuoso, cortés y caballeroso, responsable, luchador y generoso, de dos personas que merecen mi respeto más sincero, no por sus bienes o por su edad sino por su gran corazón y de quien llevo sus apellidos y nombre con gran honor, mi madre Julita y mi padre Wladimir. Este logro no es mío, es suyo, porque sé todo lo que han dejado de hacer por verme donde estoy, los amo y de verdad gracias por no dejarme caer cuando estado por dar por sentado muchas cosas, por aguantar mis mal genios, mis caprichos absurdos, por apoyarme cuando he querido empezar todo de nuevo, por compartir mis alegrías y sacarse el aire cada día para que no me falte nada. Siempre he pensado que no tengo mejores amigos aparte de mis padres, han sido siempre incondicionales, los admiro, son mi ejemplo y los amo.

También quiero compartir esta alegría con mi hermana Katy a la cual admiro y amo, que con su ejemplo desde pequeña me ha mostrado que siempre se puede salir adelante, por su alegría y hacerme ver que es lo que siempre le falta a una persona para ser luchadora lo cual es ganas de ser mejor cada día.

A mi novia Gaby que llego en el momento indicado especialmente por darme tantas risas cuando ya no he sabido que hacer con mi vida, la verdad es una mujer maravillosa y un apoyo incondicional, te amo. Gracias también a mi gran amiga Melisa.

A mi familia en general, especialmente a mis abuelitas Gloria y Eva, a mi abuelito que está en el cielo Manuel, a mi segunda madre Ximena Freire, mi tía Lourdes, mi tío Xavier que me han visto tantas veces con cara de cansancio e indirectamente me han dado fuerzas.

Creo que tenemos que agradecer tanto Juan como mi persona al excelente profesional y persona a Dr. Gustavo Chafla por incentivar nuestra creatividad en el desarrollo de esta Tesis tanto con su expertis como con recursos tecnológicos, mostrándonos siempre que un trabajo debe tener excelencia desde su principio hasta el fin. Además, sin olvidar a nuestros correctores Fabián de la Cruz, y Damián Nicolalde, que han influido nuestra vida profesional como profesores.

A mi equipo de investigación que me han apoyado como líder de proyecto siempre, y me han acolado en todas las ideas locas que se me han venido a la cabeza. Son excelentes personas como amigos. Christian un excelente programador, Bryan excelente investigador, Juanito que ha sido como un hermano en la carrera, ayudarme siempre con los servidores. Todos sin excepción han aguantado muchas cosas de mi desde enojos hasta alegrías.

Finalmente, muchas gracias a todas las personas que me han acompañado en el transcurso de esta tesis.

Diego Ponce Freire

Contenido

ÍNDICE DE TABLAS

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE DIAGRAMAS

CAPÍTULO I.

INTRODUCCIÓN

1.1. INTRODUCCIÓN

El presente proyecto describe el estudio de redes inalámbricas Wi-Fi y de Geo Posicionamiento Global GPS mediante el desarrollo de una plataforma de monitorización y localización de dispositivos móviles, cuya gestión y administración se la puede controlar desde una aplicación en entorno Web.

La plataforma será desarrollada en java con las librerías de Edición Empresarial para Web y para su publicación y consumo de servicios estará alojada en un servidor de aplicaciones Jboss.

Entre las principales funcionalidades que brindará la plataforma será el servicio de monitorización a través de GPS, adicionalmente contará con un servicio de mensajería instantánea entre usuarios dentro de una misma área definida por el cliente, y finalmente la administración y gestión de datos de usuario.

1.2. ANTECEDENTES

Cuando se habla de tecnología WI-FI, realmente se está haciendo referencia a la WI-FI Alliance. Se trata de una organización sin ánimo de lucro, que engloba a un amplio grupo de fabricantes, con el objetivo de promocionar el uso de la tecnología inalámbrica en redes de

área local, y asegurando la compatibilidad entre fabricantes en base a los estándares IEEE 802.11. La expansión de este tipo de tecnología ha sido explosiva y se prevé que en los próximos 1 ó 2 años el 90% de los equipos ya dispongan de dispositivos WI-FI. Las ventajas que ha supuesto la tecnología inalámbrica son evidentes: abaratamiento y facilidad de implantación de redes LAN, proliferación de aplicaciones y dispositivos móviles, posibilidad de crear espacios con conectividad de manera inmediata, movilidad de usuarios, etc. A toda esta funcionalidad se le suma el bajo costo de los dispositivos necesarios para su puesta en funcionamiento. (Castro, 2005)

El protocolo inalámbrico 802.11 establece varios aspectos para el acceso a las redes de áreas locales, tiene un radio que puede alcanzar los 54 Mbps en un radio de frecuencia de 5 Ghz, esta tecnología tiene la capacidad de utilizar 8 canales. La categoría B de esta tecnología tienen un acceso de hasta 11 Mbps en un radio de frecuencia de 2.5 Ghz con la capacidad de utilizar 3 canales. Estas características, funciones y capacidad de las mismas pueden depender de varios factores como el rango de cobertura, el consumo de potencia y el costo de energía implicada. En la actualidad se establece el protocolo 802.11g como el común estándar el cual permite un ancho de banda de 54 Mbps con capacidad de hasta 3 canales de 2.5 Ghz, considerando una mejora de su protocolo antecesor, de igual manera es compatible con este. (Castro, 2005)

En cambio, al tratarse de tecnología GPS se refiere a sistemas que tienen la capacidad de calcular la posición exacta de un determinado dispositivo electrónico en la tierra, inicialmente esta clase de tecnología fue usada para fines militares, sus inicios fueron a principios de los años 60’ por el departamento de defensa y transporte de los Estados Unidos.

A un inicio ya se pensaba en implementar un sistema de satélites que puedan rodear el globo terráqueo, lo cual conllevó a construir sistemas teles como TRANSIT, NAVSTAR, etc. Como alternativas para operar con dicha magnitud.

En el presente se ha desarrollado exitosamente sistemas eficaces para la administración de geo-posicionamiento los cuales ya no solo apoyan para tareas militares, sino también para administraciones empresariales, toma de decisiones financieras y económicas. Las aplicaciones que actualmente están en el mercado tienden a enfocarse principalmente a la navegación y gestión cartográfica: topografía, geodesia entre otros.

Adicionalmente este sistema ha sido integrado en los sistemas móviles principalmente en los teléfonos inteligentes, lo que ha logrado surgir un sin número de software y aplicaciones para su funcionamiento y gestión tanto para el uso personal del usuario como para posibles tareas corporativas.

1.3. PLANTEAMIENTO DEL PROBLEMA

Los sistemas de posicionamiento geográfico si bien en la actualidad están enfocados para la gestión de datos de la dicha índole como son administración de puntos geográficos, geodesia, administración de tránsito, supervisión de dispositivos, cronometría, etc.

No obstante, esta tecnología puede implementarse a un uso de enfoque empresarial y corporativo. Actualmente las empresas necesitan tomar decisiones que puedan optimizar su producción, sus ventas, formas de reducir los costos que puede conllevar a producir sus productos, venderlos y distribuirlos, para lo cual deben tomar las decisiones pertinentes para

que estos completen su proceso de venta y entrega con el costo mínimo posible. Tienden de necesidades para optimizar la supervisión del talento humano en cuanto a transporte o distribución de precios, para lo cual, es pertinente automatizar y satisfacer dichas necesidades con sistemas o aplicaciones adaptables para tal uso.

Muchas empresas o fábricas pueden tener integradas a sí mismas diferentes tipos de flotas de transporte, ya sean estas grandes o pequeñas dependiendo las rutas que puedan emplear en la entrega de su producto o para la ejecución de un servicio que ofrezcan, aquí se involucran costos como mantenimiento de los equipos de transporte, combustible, alza de precios y los impuestos que conllevan que bien pueden representar en promedio un 20 % de gastos totales empresariales.

Con las nuevas tecnologías que se nos ofrecen hoy en día es factible el hecho de minimizar los costos tanto de transporte como la supervisión y el control de recursos y talentos humanos que operan fueran de los campos y entornos de trabajo propios de la infraestructura de una empresa. Cabe recalcar entre las nuevas innovaciones herramientas tales como el GPS, el cual tiene la capacidad de permitir el control, administración y toma de decisiones a futuro para tareas y operaciones de índole logística dentro de una empresa, pues del análisis de los sistemas se puede generar reporte y estadísticas de la tasa de entrega las actividades de monitoreo de estas.

Con esta información disponible una entidad empresarial estará en la capacidad de tomar decisiones a futuro, de planificar de mejor manera sus gastos financieros, inclusive los gastos que puede conllevar los procesos macros y principales si se aplica esta tecnología para

monitorear y supervisar el personal operativo. Tomando y ejecutando decisiones pertinentes y adecuadas, el desempeño tanto de trasporte como eficiencia en la producción y gestión de productos, con las debidas decisiones, los gastos pueden convertirse en ahorros, gracias a las nuevas tecnologías de comunicación y geo posicionamiento.

El GPS junto a la conectividad de internet puede extender su funcionalidad a utilidades adicionales a la geodesia, sino también a campos corporativos y más a fondo financieros.

1.4. JUSTIFICACIÓN

Mediante el desarrollo de un sistema de monitorización de dispositivos móviles y la abstracción de esos datos, se pondrá a disposición información de geo-posicionamiento de los usuarios para el uso en diferentes áreas empresariales y domésticas.

Dentro de esta implementación cabe la necesidad del estudio de las redes en las cuales trabajan las tecnologías y servicios de internet discutidos y como se pueden integrar en el sistema de monitorización, como son: las redes WI-FI y el uso de plan de datos de dispositivos móviles. También es importante analizar las ventajas que nos brinda las redes WI-FI y como la misma identifica el posicionamiento de los usuarios que consumen los servicios de esta red, además del impacto en la autonomía de los dispositivos tecnológicos que los usuarios usan hoy en día.

Por este motivo se realizará el desarrollo de un prototipo en una plataforma web y un aplicativo en dispositivos móviles, aprovechando la evolución y mejora de las infraestructuras de las telecomunicaciones y sus beneficios de acceso a internet, como también potenciando los procesos a través de las prestaciones que la tecnología móvil que

hoy por hoy nos brinda y están a disposición de los usuarios en sus diferentes presentaciones (teléfonos móviles, tablets, teléfonos inteligentes, agendas electrónicas, computadoras portátiles, consolas de videojuegos, etc.)

Y así contar con tecnología de geo posicionamiento global, la cual puede servir a los usuarios para la toma de decisiones.

Este prototipo va a potenciarse con las librerías que Google nos brinda esencialmente la de geo-localización, las mismas pueden ser usadas en todas las diferentes plataformas de desarrollo como las que en este proyecto se pretende usar las cuales son: Java EE7, Android e IOS.

Como se mencionó anteriormente el prototipo no solo aplica a necesidades empresariales sino también a las domésticas en donde a veces es necesario el monitoreo de un grupo familiar especifico como también de grupos sociales, siempre y cuando los mismos hayan aceptado la condición de monitoreo.

Con la disposición de la tecnología GPS nos ayudará a recoger la información necesaria para que la empresa pueda tomar decisiones y realizar una planificación a futuro ya sea para su productividad de trabajo u optimización de costos y generar beneficio y valor empresarial.

1.5. OBJETIVOS

1.5.1. Objetivo General

Analizar las redes GPS y Wi-Fi y su aplicación en la monitorización mediante localización de dispositivos móviles

1.5.2. Objetivos Específicos

- Analizar el funcionamiento de las redes Wi-Fi y GPS como herramientas para la monitorización y localización de dispositivos móviles.
- Analizar las funcionalidades e integración de la librería de posicionamiento de Google para la creación de un sistema de control vía portal Web.
- Definir la funcionalidad básica para la plataforma de monitorización y localización de dispositivos móviles
- Desarrollar un prototipo funcional para la plataforma de monitorización y localización

1.6. ALCANCE

El presente proyecto de disertación culminará con la entrega de un prototipo listo para implementarlo en aplicativos móviles y un sistema de administración en ambiente web, que permitirá la entrega de datos de los dispositivos que se manifiestan en el proceso de monitoreo y localización a través de redes Wi-Fi y GPS, los mismos que serán interpretados por la aplicación y servirá para la toma de decisiones de los usuarios finales u organizaciones según sea el caso.

Además de un estudio del funcionamiento de redes Wi-Fi y GPS y como estas son usadas dentro del prototipo a desarrollarse con el Api que Google Maps nos ofrece.

1.7. FUNCIONALIDADES

1.7.1. Funcionalidades del Sistema Web

- **Acceso del Sistema:** El sistema tiene la capacidad de gestionar los usuarios intervinientes en la misma los cuales tendrán acceso para monitorear y administrar procesos y actividades del sistema web. Se inicia un proceso de registro de usuario que estará vinculado con su respectivo dispositivo cliente.

- **Mensajería Instantánea:** La aplicación web tiene como funcionalidad un módulo de chat entre los usuarios que estén registrados en el sistema ya sea a nivel de aplicación administrativa con los usuarios de la aplicación cliente.

- **Gestión de grupos:** Se podrá crear grupos entre los usuarios para generar sesiones de chat múltiples entre los mismos con el fin de clasificar usuarios sea la necesidad del administrador o entidad vinculada.

- **Monitor de Dispositivos:** La aplicación podrá visualizar en mapas digitales los dispositivos que están vinculados a los administradores y grupos que gestionan el sistema.

- **Generador de Rutas:** Una vez visualizado los dispositivos en los mapas digitales el administrador podrá generar rutas para utilidad de los usuarios clientes, así mismos estos podrán visualizarlos en sus móviles.

1.7.2. Funcionalidades del aplicativo Móvil

- **Mensajería Instantánea:** La aplicación tendrá la capacidad de enviar y recibir mensajes de texto vinculado con cualquier cliente ya sea en una aplicación web o móvil que este registrado en su respectiva lista de contactos.

- **Visualización y envío de ubicación GPS:** La aplicación podrá enviar ubicaciones por medio del sistema GPS del dispositivo móvil para que estos sean visualizadas y administradas por la aplicación administradora web.

1.8. METODOLOGÍA DE DESARROLLO

Metodología para el Desarrollo de Software: (XP) Extreme Programming.

XP es una metodología enfocada directamente con el cliente donde cada iteración es una versión mejorada del software con respecto a la información y retroalimentación del cliente con la finalidad de la calidad del software y la satisfacción del anterior mencionado.

1.8.1. Fase de Exploración

En esta fase se determina las historias del usuario, las cuales servirán de módulos de programación para el equipo de desarrollo.

1.8.2. Fase de Planteamiento

Se establece el plan para las entregas o puntos de control dentro del desarrollo del proyecto, en forma de plan de iteraciones.

1.8.3. Fase de Diseño

Se genera todos los diagramas necesarios dentro del proyecto

1.8.4. Fase de Codificación

Se realiza la implementación de código para cada módulo o avance establecido dentro del plan de iteraciones.

1.8.5. Fase de Pruebas

Fase aplicada a la finalización y aprobación de cada avance dentro del plan de iteración.

1.8.6. Fase de Implementación

La aplicación será puesta a pruebas dentro de entornos reales instalándola en un servidor dedicado a este fin, y en los dispositivos clientes que sea necesario.

1.9. ESTUDIO DE FACTIBILIDAD

Se dispone de los recursos tecnológicos para el desarrollo del software del proyecto los cuales son los siguientes:

1.9.1. Factibilidad Técnica del proyecto de desarrollo de Software

1.9.1.1. Software

Tabla I-1: Factibilidad Software para el proyecto de Desarrollo de Software

No.	Descripción	Aplicación de SW utilizado
1	IDE de Desarrollo Aplicación Web	Eclipse 4.6 Neon para Java EE
2	IDE de Desarrollo Cliente	Android Studio 1.5.1
3	Servidor de Aplicaciones	Jboss AS 7.1
4	Motor de Base de datos	My SQL CS GPL 5,5,45
5	Framework de Desarrollo	JSF Prime Faces 6 RC4
6	Navegador para Pruebas	Mozilla Firefox 42.0
7	SO. App Cliente	Android Lollipop 5.1

Elaborado por: Diego Ponce – Juan Prado

1.9.1.2. Hardware

Tabla I-2: Factibilidad Hardware para el proyecto de Desarrollo de Software

No.	Descripción	Cantidad	HW utilizado
1	Computadora de Escritorio	2	Procesador: Intel Core i7 RAM: 4GB HDD: 1TB SO: Win 10 Pro
2	Computadora de Escritorio (Servidor BD y de Aplicaciones)	1	Procesador: Intel Core 2 Duo RAM: 2GB HDD: 500 GB SO: Win Server 2012
3	Dispositivo Móvil	1	Sony Experia Z3 Android 5.1

Elaborado por: Diego Ponce – Juan Prado

1.9.2. Factibilidad Económica de proyecto de desarrollo de Software

Tabla I-3: Factibilidad Económica para el proyecto de Desarrollo de Software

No.	Descripción	Cant. / Mes	Costo
1	**Recursos Talento Humano**		
2	Desarrollador 1	3	$600,00
3	Desarrollador 2	3	$600,00
4	**Recursos HW**		
5	Computadores de Escritorio	3	$4.000,00
6	**Recursos SW**		
7	Eclipse 4.6 Neon para Java EE	2	$0,00
8	Android Studio 1.5.1	2	$0,00
9	Jboss AS 7.1	1	$0,00
10	My SQL CS GPL 5,5,45	1	$0,00
11	JSF Prime Faces 6 RC4	1	$0,00
12	Mozilla Firefox 42.0	1	$0,00
13	Android Lollipop 5.1	1	$0,00
14	**Otros**		
15	Servicio Básicos	3	$80,00
16	Transporte	3	$20,00
17	Servicio de Internet	3	$0,00
18	Papelería	3	$10,00
	TOTAL		**$5.310,00**

Elaborado por: Diego Ponce – Juan Prado

Los gastos antes estipulados son asumidos en su mayoría por la PUCE, los gastos como la categoría de “Otros” han sido asumidos por los titulares del trabajo de disertación, con todos los rubros antes detallados queda concluido que el desarrollo de este proyecto es factible económicamente.

1.9.3. Factibilidad Operativa del proyecto de desarrollo de Software

El estudio de las redes GPS y Wi-Fi mediante el desarrollo del software de monitoreo y localización de dispositivos móviles se ha desarrollado en lenguaje Java 2 E.E. con JSF, este hecho permite a la aplicación web conectarse con diferentes motores de base de datos los mismos que pueden ser gestionados con gran facilidad por varios IDE's de libre uso.

Así mismo la amplia compatibilidad con la el Motor de MySQL 5.5 junto a su conexión, configuración en código fuente no ha presentado mayor problema al desarrollar los diferentes módulos CRUD del sistema web.

El desarrollo de la aplicación cliente se lo ha generado en base a un sistema operativo estable y compatible con sus versiones tanto antecesoras como predecesoras, no obstante, el comportamiento del mismo puede cambiar conforme varíen las características y prestaciones del hardware del dispositivo, las funcionalidades del sistema operativo y las funciones de las API's utilizadas para el mismo.

La comunicación entre la aplicación web con los diferentes dispositivos clientes se ha resuelto de dos maneras, la parte de la gestión de datos, en los módulos de gestión de datos CRUD y administración de la geolocalización se maneja servicios web REST, los cuales son consumidos por los dispositivos clientes, y así mismo como retro alimentación devuelven datos que serán guardados en un servidor de BD. El módulo de mensajería instantánea está resuelto por las librerías de GCM (Google Cloud Messaging), el cual permite un servicio de chat óptimo para este tipo de sistemas en donde se requiere seguridad, integridad y alta disponibilidad de la información.

También se toma en cuenta el uso del hardware como los dispositivos GPS y WI-FI existentes en un dispositivo móvil, sin dejar a tras las demás interfaces NIC que puedan

existir en ellos, concluyendo así que se ejecutará en todo dispositivo que tenga la capacidad de conectarse a internet.

Conforme la tecnología siga evolucionando tanto en software como en hardware, los dispositivos móviles obtendrán más potencia en cuanto a la precisión de ubicación mediante sistemas GPS, la intensidad de señal tanto WI-FI como de los demás medios para conectarse a internet, etc. Por este hecho se concluye que el sistema no quedará obsoleto en muy poco tiempo, al contrario, este software ha sido desarrollado con un ideal evolutivo, tal que pueda seguirse adaptando conforme a las necesidades tanto de cliente como de empresas.

CAPÍTULO II.

MARCO TEÓRICO

Desde el inicio, la creación del internet ha causado de forma masiva que la información necesaria para un determinado usuario esté siempre disponible y al alcance de sus manos para que pueda ser visualizada, no suficiente, la tecnología dio paso a la interactividad entre la información guardada en la nube y el cliente que lo requería, mediante formularios y objetos que tienen capacidad de escuchar eventos generados por el usuario.

Conforme la tecnología y el internet avanzaron se creó la posibilidad gestionar datos almacenados en servidores, almacenar multimedia e inclusive realizar estadísticas con esta información, así mismo las interfaces usuario-maquina siguieron su respectivo proceso evolutivo, la conectividad por internet vía modem telefónico se dejó atrás por conexiones más estables y rápidas como el ADSL[1] y así mismo la comunicación con sus máquinas consumidoras, tal como son las conexiones vía cable UTP[2], conexiones inalámbricas WI-FI, etc., (Niola, 2015).

Finalmente, el hardware que brinda funciones complementarias a la tecnología digital antes mencionada fueron tomando avances considerables dentro del campo de la gestión de la información digital y el internet, dispositivos inalámbricos, sistemas de posicionamiento y monitoreo geográfico, cámaras multimedia integrado en los dispositivos móviles, este hecho ha provocado la creación de aplicaciones gestoras de todo tipo de información que

[1] **ADSL:** Bucle de Abonado Digital Asimétrico, Protocolo para servicio de Internet.
[2] **UTP:** Cable de par trenzado no blindado.

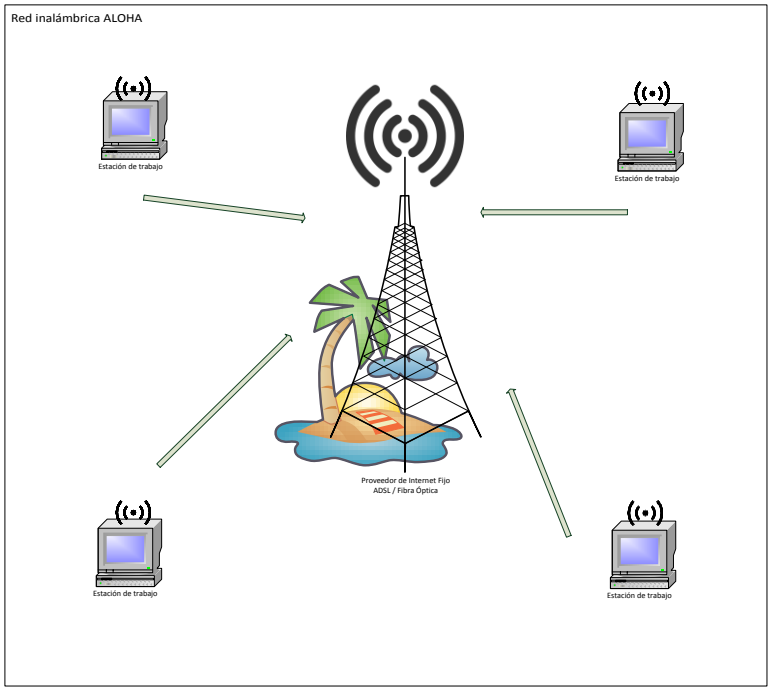
aprovechen e integren todo este tipo de hardware y funcionalidades de todo dispositivo que pueda conectarse a internet.

2.1. REDES INALÁMBRICAS WI-FI

Las tecnologías inalámbricas fueron remontadas por el año de 1971 en una investigación bajo el mando de Norman Abramson en la universidad de Hawaii, en donde se logra crear el primer motor de gestión y trasmisión de paquetes mediante un receptor y trasmisor por radio, dicha conjunto de dispositivos en red se llamó "ALOHA" la cual se consideró la primera red local inalámbrica WLAN[3], posteriormente solucionaron el problema de acceso a la dirección física (MAC)[4], trasmitiendo frecuencias diferentes a dichos dispositivos (Universitat Politécnica de Valencia, 2010).

La infraestructura de dicho proyecto puede ser representada en la Ilustración II-1.

[3] **WLAN:** Wireless Local Area Network, Red para comunicación de dispositivos inalámbricos.
[4] **MAC:** Control de Acceso al Medio, identificador para dispositivos NIC de comunicación.



,

Ilustración II-1: Arquitectura del Proyecto ALOHA

Elaborado por: Diego Ponce – Juan Prado

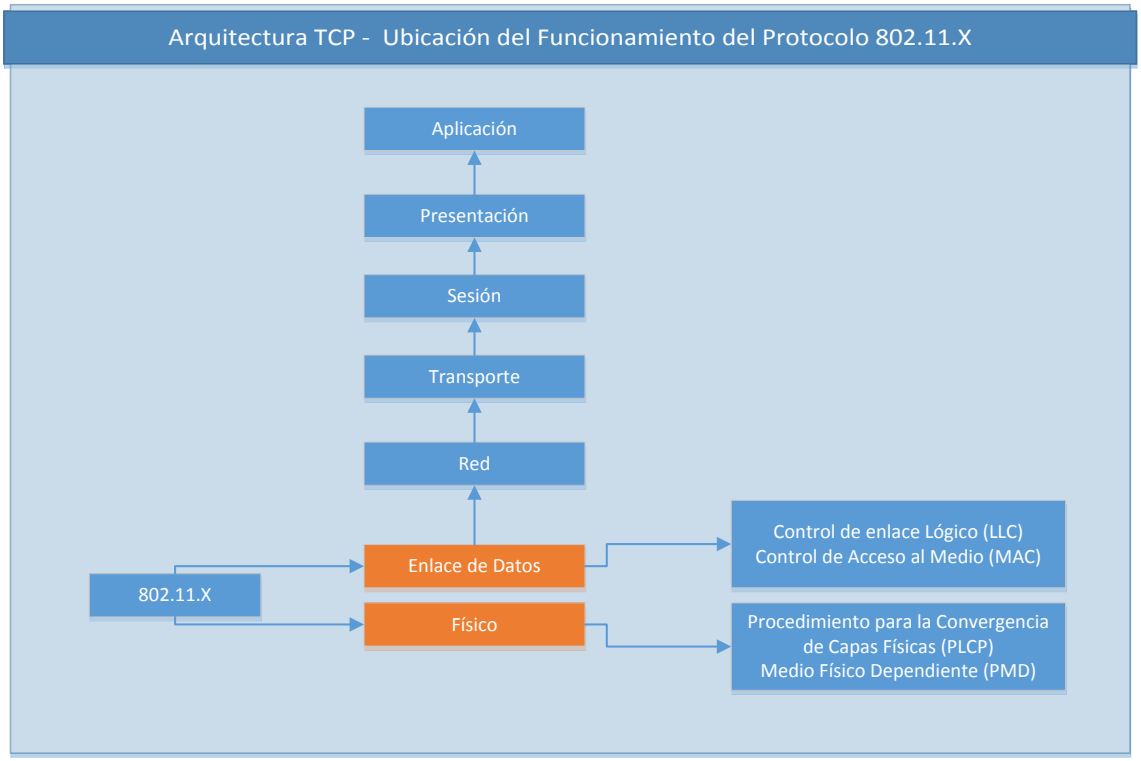
Basado de: (Universitat Politécnica de Valencia, 2010)

Posteriormente el proyecto ALOHA[5] se implementó al proyecto estadounidense ARPANET[6] creado por el departamento de defensa de los EE. UU para la gestión de la comunicación de los diferentes departamentos gubernamentales de dicha nación.

Las redes Wi-Fi es parte de las tecnologías inalámbricas para la comunicación y transferencia de datos comúnmente utilizada en el manejo y acceso a internet, cualquier dispositivo que cuente con dicha tecnología puede comunicarse entre sí, sin distinción de su marca, solo es necesario que trabaje con el estándar IEEE802.11[7].

[5] **ALOHA:** Primer proyecto para sistema de redes de computadora.
[6] **ARPANET:** Red de agencia de proyectos de Investigación Avanzada.
[7] **IEEE 802.11:** Estándar para la definición de la infraestructura de la capa física de Datos OSI.

"El estándar 802.11 presenta la misma arquitectura de toda la familia 802.x y esta referenciada al modelo OSI enfatizando la separación del sistema en dos partes principales; la capa de enlace de datos (DLL) Mac, y la PHY[8] (capa física) " (Niola, 2015). Como es



mostrada en la Ilustración II-2.

Ilustración II-2: Estructura del funcionamiento OSI con el protocolo 802.11.X

Elaborado por: Diego Ponce – Juan Prado

Basado de: (Niola, 2015)

El funcionamiento de las redes Wi-Fi está basado básicamente en información trasmitida mediante ondas electromagnéticas de punto a punto, mismas señales contienen una frecuencia intensa más alta que la frecuencia modular, estas son conocidas como ondas portadoras.

[8] **PHY:** Se refiere a una interfaz entre la dirección MAC y el su entorno inalámbrico.

La onda moduladora tiende a acoplarse a su complemento portador, el radio que ocupa estas frecuencias es conocido como el ancho de banda debido al acoplamiento de las dos primeras.

La IEEE 802.11 antes mencionada lanzó dos tipos de funcionamiento en cuanto corresponde a la operación de las redes Wi-Fi las cuales son Infraestructura y Ad-hoc[9], en estas la primera y la comúnmente utilizada en equipos u ordenadores que se comunican a uno o más puntos de acceso los cuales están conectados mediante un cable de datos a una red del mismo tipo, es decir, una red cableada, la segunda se trata de dispositivos que están comunicados independientemente entre sí.

2.1.1. Tipos de Redes Wi-Fi

Las redes inalámbricas fueron pensadas para dar el paso de reemplazar recursos de hardware como es el cableado, par trenzado, fibra óptica, cable coaxial, etc.

En la actualidad la tecnología puede brindarnos algunos tipos de redes inalámbricas como las siguientes.

1. **Red inalámbrica de ámbito Local (Wireless Local Area Network-WLAN)**, son redes que cubren áreas relativamente pequeñas o de un contexto doméstico, ya sean en casas, oficinas de trabajo pequeñas, PYMES, etc.

[9] **Ad-hoc:** Grupo de red Inalámbrica no centralizada, esta no depende de un dispositivo puente.

2. **Red inalámbrica de ámbito Personal (Wireless Personal Area Network-WPAN)**, las cuales han sido creadas para el funcionamiento único de un usuario, quiere decir, un área realmente reducida, puede ponerse como referencia una sola habitación, en un principio esta tecnología empezó con hardware de infrarrojos, en el presente ha sido reemplazado por tecnología inalámbrica Bluetooth.

3. **Red inalámbrica de área extensible (Wireless Wide Area Network-WWAN),** son redes cuya cobertura tienen objetivo áreas realmente extensas como pueblos o ciudades enteras, estas ya pueden ser usadas por compañías grandes proveedoras de internet como son las ISP's.

2.2. SISTEMA DE POSICIONAMIENTO GLOBAL (GPS)

"*El sistema de posicionamiento global, GPS, es un sistema mundial de navegación desarrollado por el Departamento de Defensa de los Estados Unidos. Actualmente este sistema consta de 24 satélites artificiales (21 regulares más 3 de respaldo) y sus respectivas estaciones en tierra, proporcionando información para el posicionamiento las 24 horas del día sin importar las condiciones del tiempo"* (Barry F. Kavanah, Gelnn Bird J.S, 1989).

Este sistema funciona a base de una red de satélites que fueron puestos en órbita a partir del año de 1973 y su lanzamiento fue en la misma década cinco años después, no fue hasta 1995 que se consideró el proyecto en plena operatividad para uso exclusivamente militar en funciones como la gestión de información geodésica y topográfica.

Adicionalmente estos sistemas obtienen su posición basándose de la medición de distancias de por lo menos tres diferentes satélites a un determinado sitio o punto sobre la superficie de la Tierra, se puede lograr la posición de la misma por Trilateración[10] (M., 2002).

La Infraestructura de los satélites está representada en la Ilustración II-3.

Ilustración II-3: Representación gráfico de satélites GPS alrededor de la Tierra

Elaborado por: Diego Ponce – Juan Prado

El sistema GPS ha tenido diferentes usos y aplicaciones, entre ellas se puede encontrar las siguientes:

- Gestión y cálculo de posicionamiento Satelital.
- Medición de distancia por parte de los Satélites.
- Medición y gestión del tiempo.
- Gestión de la órbita del satélite.
- Gestión y correcciones de ondas.

2.2.1. Sistema de Geo Posicionamiento por Wi-Fi

[10] **Trilateración:** Método del área matemático utilizado para calcular posiciones de objetos en una perspectiva relativa.

Si bien los dispositivos móviles hoy en día cuentan con el hardware y sistemas de posición por GPS, hay que tomar en cuenta ambientes privados de acceso a internet, por ejemplo, la ausencia de acceso a datos móviles por falta de cobertura de conexión o saldo crédito por parte de la cuenta móvil del usuario.

Con el registro cartográfico detallado de calle, avenidas y multimedia de los mismos por parte de google se ha implementado adicionalmente el registro de posición por hardware por puntos de acceso a internet, su funcionamiento se basa en la siguiente manera.

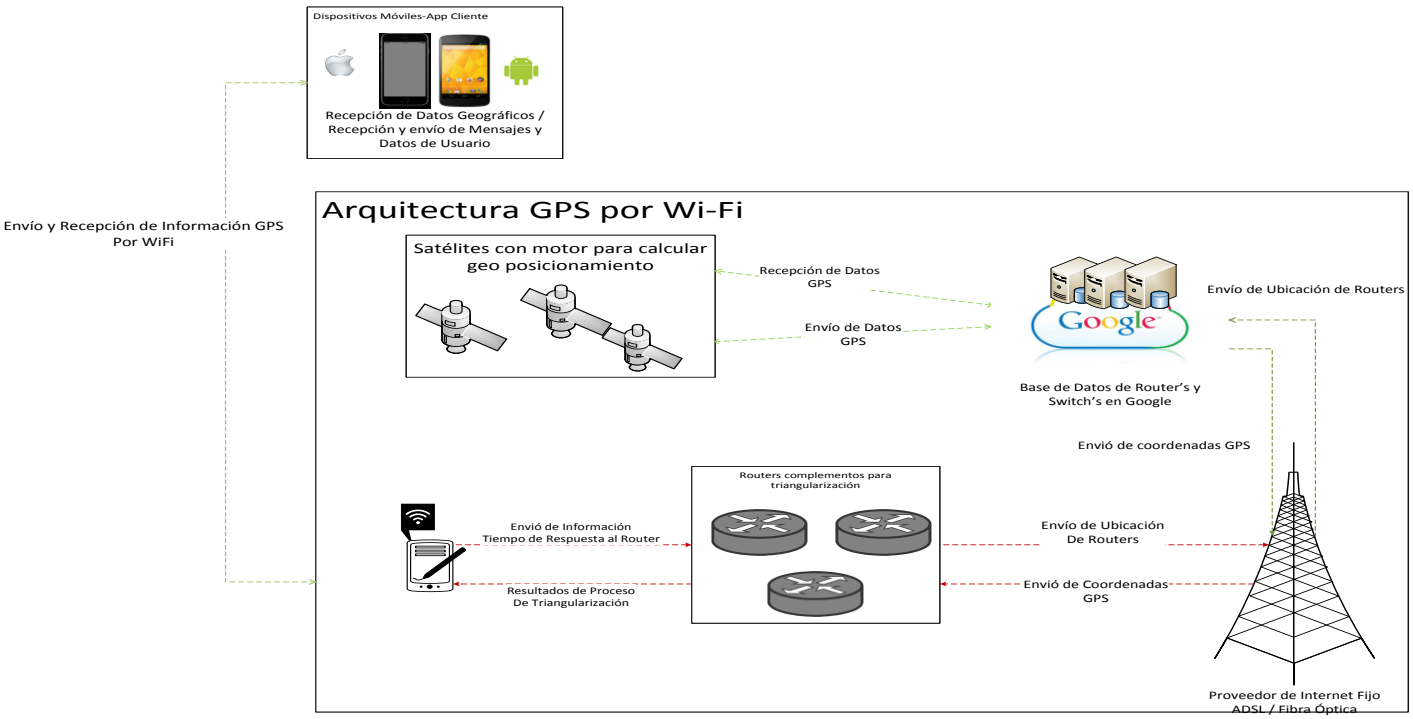
1. Google toma la información de los puntos de acceso Wi-Fi y la almacenará en una base de datos dedicado al almacenamiento de puntos de acceso a la nube.

2. El móvil o dispositivo que necesita ubicación no necesariamente debe identificarse e iniciar sesión con el punto de acceso para poder determinar una ubicación. Sea el caso de serlo así solo obtendrá datos como identificación del ruteador y sus coordenadas obtenidas de la nube de Google, obviamente este se conectará a dicha base de datos y obtendrá una latitud y longitud.

3. Para que la precisión de la ubicación sea medianamente precisa, al menos debe hacerse el mismo proceso anterior con tres ruteadores, o puntos de acceso, de esta manera el margen de error de la ubicación, será menor.

4. Se promedia la ubicación de los ruteadores usados para determinar la posición del dispositivo y la refleja en su respectiva aplicación de geo posicionamiento.

En la Ilustración II-4 se aprecia la infraestructura y funcionamiento para



determinar la posición de un dispositivo móvil o cliente.

Ilustración II-4: Arquitectura del funcionamiento de Geo Posicionamiento por Wi-Fi

Elaborado por: Diego Ponce – Juan Prado

2.3. APLICACIONES WEB

Desde la creación y lanzamiento de la web 2.0 han surgidos varios prototipos de aplicaciones web, con el principio de la implementación de formularios dentro del tradicional código HTML hasta los grandes sistemas de información en línea que podemos apreciar en todo el internet.

"Las aplicaciones web permiten la generación automática de contenido, la creación de páginas personalizadas según el perfil del usuario o el desarrollo del comercio electrónico. Además, una aplicación web permite interactuar con los sistemas informáticos de gestión de

una empresa, como puede ser gestión de clientes, contabilidad, inventario, a través de una página web." (Mora, 2002).

La infraestructura de las aplicaciones web se basa en la arquitectura de aplicaciones Cliente-Servidor, en este entorno un ordenador cliente donde está alojada una aplicación del mismo tipo enviará información o requerimientos de la misma. El Servidor escucha, espera y recibe solicitudes o peticiones de información del cliente, el cual no necesariamente tiene que estar en su mismo entorno tanto de plataforma como de comunicaciones.

2.3.1. Servidor de Aplicaciones

Todo este entorno de sistema y aplicaciones deberá estar alojados en servidores de aplicación, el cual tiene como función principal el de cargar en memoria todas las instancias y componentes necesarios para el adecuado funcionamiento de la aplicación que será lanzada de forma distribuida.

"El servidor debe encargarse de la creación y de la carga en memoria de las instancias y los componentes, así como la gestión de una cola de espera para satisfacer las peticiones de los clientes. Además, para satisfacer las exigencias de las aplicaciones corporativas, el servidor de aplicaciones debe ser potente y fiable. Es por tanto capaz de gestionar la disponibilidad (escalabilidad) de las aplicaciones (balanceo de carga, tolerancia a fallos) utilizando grupos (Clúster)[11] *de servidores* (AUMAILLE, 2002).

2.3.2. Funcionamiento del Servidor de Aplicaciones

[11] **Clúster:** Es un conjunto de ordenadores aglomerados en una infraestructura tecnológica Determinada.

Los servidores de aplicaciones en su instalación física, y en un mismo ordenador del mismo tipo es adaptable y configurable en un entorno de más servidores de aplicación lógicos. Cada unidad lógica puede componerse por dos unidades: un módulo de contenido web y un contenedor de gestión de datos y acceso a datos, por ejemplo, una clase EJB[12], el rol de estos contenedores es poner a disposición de los dispositivos clientes los servicios que les corresponden.

"Un servidor de aplicación está compuesto además de un conjunto de servicios (no representados en el esquema) útiles para el funcionamiento de los contenedores y componentes. Uno de los más importantes es el servicio de nombres JNDI[13], que permite al servidor de aplicación o a un cliente conocer la máquina física sobre la que se encuentra un recurso Web, una fuente de datos o el contexto de transacción distribuida, para que así puedan tener acceso al mismo." (AUMAILLE, 2002).

Dentro de las distintas plataformas para el desarrollo de aplicaciones web para el desarrollo del presente proyecto se enfocará en el lenguaje J2EE (Java Edición Empresarial) y el motor para servidor de aplicaciones JBOSS[14].

2.3.3. Lenguaje de Programación J EE (Java Enterprise Edition) (Back-End)

Java EE fue desarrollado por la corporación de Sun Microsystems. La plataforma fue lanzada al final de la década de los 90's con su segunda edición (J2EE). Esta caracterizado

[12] **EJB:** Enterprise Java Bean, Interfaces para el desarrollo de aplicaciones en Java EE.
[13] **JNDI:** Interfaz para el desarrollo de programación para directorio y funcionalidades de aplicaciones en Java.
[14] **JBoss:** Motor para servidor de aplicaciones especializado en Java EE.

por proporcionar técnicas de desarrollo empresarial en cuanto se refiere a aplicaciones distribuidas, de proporciones robustas y de alta información e íntegra.

Derivado de su predecesor Java, es un lenguaje cuyo entorno de ejecución tiene la capacidad de ser multiplataforma y ha sido acogido por sus características que contiene tanto como su portabilidad e independencia como fue mencionado anteriormente.

Java como su lenguaje nativo conjunto a la plataforma de edición empresarial brindan un apoyo original, eficiente e íntegro para la creación y el desarrollo de aplicaciones distribuidas y aplicaciones web, las cuales pueden adaptarse a tecnologías actuales, permitiendo así, acoplarse a los sistemas tanto existentes en el presente como los que ya han sido descontinuados (AUMAILLE, 2002).

2.3.3.1. APIs o componentes que conforman J EE

- **Servlet:** Esta API contiene y dispone de todos los componentes necesarios para el desarrollo y funcionamiento de elementos web dinámicos en Java.

- **JSP (Java Server Page):** Es una herramienta para el diseño de páginas web interactivas con el usuario final, las cuales contiene sentencias y comandos nativos de java que pueden ser embebidos en un cuerpo de página tradicional HTML.

- **JPA (Java Persistence API):** Este módulo de extensión proporciona un servicio de mapeo de la base de datos convirtiéndolos en objetos relacionales (ORM)[15]. Adicionalmente contiene el llamado (JQPL)[16] que define al sistema de

[15] **ORM:** Mapeo de objetos relacionales, método de desarrollo para vincular entidades objetos y tablas de una base de datos.
[16] **JQPL:** Lenguaje de Consultas para Persistencia en Java.

lenguaje de consultas de Java que puede indexar información contenida en la base de datos vinculados a los objetos mapeados por este servicio.

- **EJB (Enterprise Java Bean):** Este componente está encargado de brindar un diseño de aplicaciones las cuales son basadas en arquitecturas de desarrollo de programación en "N" capas. Adicionalmente, La API[17] dispone de varios servicios para el desarrollo de aplicaciones como los de persistencia, transacción, cuya gestión de información está exclusivamente manejada por el EJB.

- **RMI/IIOP (Remote Method Invocation/Internet Inter-Orb Protocol):** Esta API tiene la capacidad del diseño de aplicaciones distribuidas en el lenguaje de Java, también permite a aplicaciones RMI[18] la interacción con aplicaciones CORBA[19].

- **JMS (Java Message Service):** Este componente permite la disponibilidad y acceso a los servicios de un motor de mensajería instantánea para la gestión asíncrona de llamadas a los componentes e instancias básicas que puede conformar una aplicación desarrollada en Java E.E.

- **Extensión de JDBC 2.0 (Java DataBase Conectivity):** Este Plugin o extensión adicional de la plataforma tiene la función de comunicación con fuentes de datos externos mediante una conexión JDBC, la misma fuente de datos estaría

[17] **API:** Interface para Programación de Aplicaciones, conjunto de funciones y rutinas para extensión de bibliotecas de programación.
[18] **RMI:** Método para Invocación remota de Java, método para la comunicación en desarrollo en ambientes de servidor.
[19] **CORBA:** Arquitectura de rompimiento para solicitudes de objetos comunes, tiene la capacidad de relacionar varios objetos creados en diferentes plataformas de desarrollo.

ocupándose de diferentes conexiones al mismo tiempo. Esta trasmisión de datos está basada en otro componente de Java E.E. conocido como JTA.

- **JTA (Java Transaction Api):** JTA se encarga de gestionar las transacciones y paso de datos dentro de la infraestructura de una aplicación web, su funcionamiento se basa en la distribución de transacciones entre los diferentes EJB respectivamente, así mismo si es necesario la comparte entre las distintas conexiones de fuentes de datos externas.

- **JNDI (Java Naming and Directory Interface):** Dispone de acceso a servicios dentro de la plataforma como son de tipo nombre o índice, este será proporcionado a la interfaz SPI (Proveedor de servicios de Interfaces).

- **Java Mail:** Esta herramienta brinda características y funciones de correo electrónico dentro del desarrollo de la aplicación

2.3.3.2. Servicios de JEE

- **Servicios de Nombres:** De todos los servicios que puede contener un servidor de aplicaciones el más destacable es el de Nombres, pues dispone de almacenamiento en cuanto a los recursos de objetos o instancias como son:
    - Componentes Ejb.
    - Conexión externa a datos (JDBC)[20].

[20] **JDBC:** Librería para entorno de ejecución de consultas SQL.

- Fuentes de Datos (Data Source).
- JTA,
- Servicio de mensajería JMS.
- Servicio de Configuración.

Adicionalmente, esto lo hace de forma jerárquica, de tal manera que todos los dispositivos clientes puedan solicitar estos servicios y utilizar todo tipo de recursos implementados en la aplicación web y comunicarse remotamente.

Los servicios de nombres tienen características que brindan motores de búsqueda, alcance y ubicación de recursos como es el SPI (Servicio de proveedor de Interfaces) y el protocolo LDAP (Lightweight Directory Access protocol)[21] (AUMAILLE, 2002).

- **Servicio de gestión de las transacciones: "***El gestor de transacciones es una implementación de la API JTS. Permite gestionar transacciones locales, pero, sobre todo, se encarga de la gestión de las transacciones distribuidas, indispensables para las aplicaciones J2EE. La misma transacción puede ser compartida por varios componentes y por varias fuentes de datos."* (AUMAILLE, 2002).

Toda transacción puede estar disponible para varias fuentes de datos, así mismo para varios componentes e instancias. El funcionamiento tiene la responsabilidad que las propiedades ACID (Atómico, consistente, durable, aislado), siempre sean estándar dentro del proceso de transacción.

[21] **LDAP:** Adicional, este permite el acceso a un recurso en la red.

Adicional a esto, el proceso de transacción debe realizar ya sea la validación o la denegación de la misma tomando en cuenta el protocolo de verificación que se conoce en dos fases 2PC (2 Phase Commit).

- **Servicio de gestión de la disponibilidad de las aplicaciones (Balanceo de carga, tolerancia a fallos):** Un servidor de aplicaciones como tal tiene la responsabilidad de tener control y gestión de todos los accesos y peticiones por parte de los clientes que necesitarán acceso a la información a los servicios de la aplicación web, así mismo mantener sesiones y correspondientemente dar los resultados simultáneos, invocar las veces que sea necesario un único recurso para satisfacer las necesidades de los dispositivos clientes antes mencionados en un tiempo mínimo y el cual pueda reflejar eficacia e integridad de la aplicación.

"Para ello, el servidor de aplicación utiliza un proceso de creación de una hebra (thread[22]) asociada a una respuesta, lo que permite ejecutará el recurso en un entorno multitarea. Además, para mejorar notablemente las prestaciones, el administrador puede utilizar un clúster (grupo) de servidores que participan en el funcionamiento de la aplicación." (AUMAILLE, 2002).

El servidor de aplicaciones debe estar encargado de tener siempre balanceado la carga de información y peticiones que puede tener conjuntamente con todos los servidores dentro de su entorno sea el caso. Adicionalmente, debe gestionar y ser tolerantes a los diferentes fallos que puedan presentarse, de esta manera todos los procesos y solicitudes no serán interrumpidas en cambio serán redirigidas al resto de servidores dentro del entorno o el clúster.

[22] **Thread:** Flujo o canal de funcionamiento de una aplicación informática el cual se encuentra en un entorno de ejecución.

- **Servicio de Seguridad:** Son herramientas adicionales para la gestión de grupos de usuarios o configuración de los siguientes dominios para los diferentes perfiles y accesos que se necesite para realizar la asignación de permisos, los mismos que posteriormente afectarán cada elemento, instancia y recurso de información dentro de la aplicación web.
- **Servicio de Administración:** Los diferentes motores sistemas y plataformas dentro del servidor de aplicaciones contiene una consola ya sea gráfica o mediante línea de comandos.

"La consola de administración representa de forma gráfica, en forma de representación absorbente, los datos del dominio administrativo del servidor de aplicación. Con esta vista es posible consultar, modificar y añadir elementos de la configuración de las aplicaciones corporativas y del servidor de la aplicación" (AUMAILLE, 2002).

- **Servicio de acceso a datos:** Implementación que tiene la capacidad de configurar fuentes externas de datos y desplegar las mismas en conjunto con el controlador del servidor de aplicaciones y el servidor de Base de datos.

2.3.3.3. Relaciones entre los tipos de clientes y las aplicaciones distribuidas en JEE

- **Cliente Ligero:** Los cuales tienen acceso comúnmente desde navegadores web. Su funcionamiento es realizando una solicitud mediante protocolo HTTP[23], si el elemento o recurso solicitado es un objeto estático se retornará por el mismo medio de canal de HTTP.

[23] **HTPP:** Protocolo de Transferencia de Hiper Texto, es decir recursos cargados en internet.

Caso contrario, si el recurso solicitado es un objeto dinámico, se filtrará la solicitud dentro del servidor de aplicaciones y por ende también del servidor de base de datos o la fuente externa de la misma, este recurso se procesa para ser enviado de vuelta al cliente por el mismo protocolo HTTP.

- **Cliente Pesado:** Mediante una aplicación de escritorio comúnmente previo instalación, la cual puede estar conformada mediante Servlets[24] y JSPs[25] que usarían un protocolo HTTP al igual que los clientes ligeros, mayormente cuentan con una interfaz gráfica y un gestor de eventos, no obstante, ya es un tipo de cliente que no se usa para el desarrollo pues su producción es independiente para computador, en cambio con el cliente ligero su despliegue es desde el servidor de aplicaciones o el host donde está alojado la página web.

2.3.4. Arquitectura del Servidor de Aplicaciones

"La mayor parte de los servidores de aplicación están compuestos de un contenedor Web y de un contenedor EJB. Esto incluye a los servidores comerciales IBM WebShepere Application Server y a BEA Web Logic. Sin embargo, podemos encontrar en el mercado soluciones Open Source que ofrecen tan solo uno de los dos tipos de ordenador, como el contenedor web Apache TOMCAT[26] o el contenedor de EJB Evidian JOnAS" (AUMAILLE, 2002).

[24] **Servlet:** Componentes de desarrollo los cuales extienden las fucionalidadaes de los servidores web.
[25] **JSP:** Java Server Pages, plataforma o tecnología de desarrollo Web 2.0.
[26] **Tomcat:** Plataforma para contenido de Servlet.

2.3.4.1. Servidor de Aplicaciones JBOSS

"Jboss es un servidor de aplicaciones JEE de código abierto implementando en Java puro. Al estar basado en Java, JBoss puede ser utilizado en cualquier sistema operativo que lo soporte." (Sánchez, 2007).

JBoss al ser conocido como el primer servidor desarrollado en código abierto y su lenguaje nativo como Java, tiene completa capacidad para el soporte de la edición empresarial del lenguaje antes mencionado, su rendimiento está optimizado para trabajar con aplicaciones empresariales y soportar altos índices de solicitudes y manejo de procesos.

Este servidor de aplicaciones contiene características como las siguientes:

- Confiabilidad para empresas.
- Flexibilidad.
- Servicios Middleware para objetos e instancias propios del Lenguaje Java.
- Estándares puntuales.
- Plataforma con licencia de código abierto y libre uso.
- Soporte con disponibilidad completa del código fuente.
- Soporte JMX[27].

[27] **JMX:** Administrador de Extensiones de Java.

2.3.5. Plataforma de Desarrollo Web PrimeFaces

En la actualidad muchas empresas requieren tecnología de última generación y aplicativos ideales para la satisfacción de sus necesidades con respecto al negocio del mismo. Los nuevos avances tecnológicos nos ofrecen varias plataformas de desarrollo para empresas las cuales son mucho más escalables robustas y tolerables a fallos permitiendo así un alcance del rendimiento mucho más alto en comparación a lo que era la Web 1.0.

PrimeFaces como tal es un Framework[28] de código abierto a modificaciones para su nativa plataforma Java Server Faces 2.0, esta tecnología es desarrollada por la corporación Prime Technology, la cual tiene como misión proveer de herramientas y técnicas siempre a la innovación del diseño web y aplicativos del mismo ámbito.

Los componentes de dicha plataforma están soportando actualmente AJAX[29] para tener un control adecuado de eventos entre los recursos y peticiones que se maneja dentro del aplicativo a desarrollar, adicional que cuenta con su propio WebKit[30] para el desarrollo de aplicaciones web y el diseño responsivo de los mismos (Fernando Pech-May, Mario A. Gomez-Rodrigez, Luis A de la Cruz-Diaz, Salvador U. Lara-Jerónimo, 2010).

"Algunas de las nuevas tecnologías que han surgido son: JavaServer Faces (JSF) que es la tecnología estándar de la edición empresarial de Java (Java Enterprise Edition, Java EE), para la creación de interfaces de usuario en la web y que permite integrar otras tecnologías como hojas de estilo en cascada (CSS) que describe cómo se va a mostrar un documento, Ajax Asynchronous Java Script and XML); un modelo de desarrollo web para crear

[28] **Framework:** Plataforma o entorno para ejecutar o desarrollar actividades determinadas o aplicaciones en un lenguaje predeterminado.
[29] **AJAX:** Motor de JavaScript y XML Asíncrono, método para desarrollo aplicaciones interactivas.
[30] **WebKit:** Plataforma para herramienta desarrollo de aplicaciones para navegadores web.

aplicaciones interactivas, JavaBeans empresariales (Enterprise JavaBeans, EJB) y el API (Application Programming Interface) de Java para el maneo de entidades persistentes (Java Persistence API, JPA) sobre base de datos relacionales" (Fernando Pech-May, Mario A. Gomez-Rodrigez, Luis A de la Cruz-Diaz, Salvador U. Lara-Jerónimo, 2010).

En la Ilustración II-5 se muestra como las tecnologías de Java Edición Empresarial pueden ser utilizadas en entornos distribuidos y web:

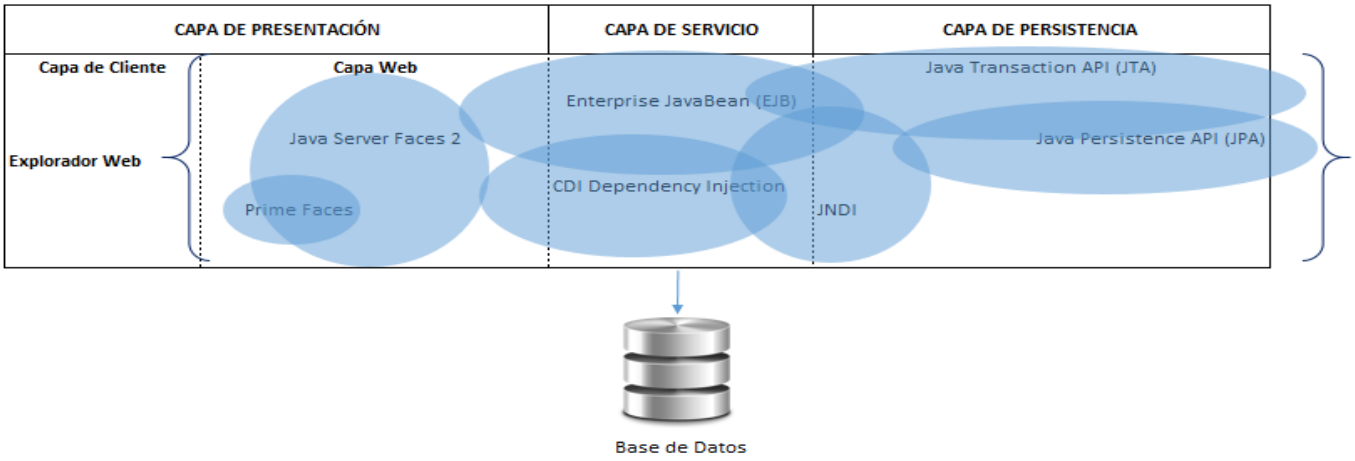


Ilustración II-5: Arquitectura de tecnologías de Java 2EE

Elaborado por: Diego Ponce – Juan Prado

Basado de: (AUMAILLE, 2002)

2.4. APLICACIONES MÓVILES

Conforme la tecnología ha ido avanzando, paralelamente han cambiado, las técnicas, paradigmas y herramientas para el desarrollo de aplicaciones, cada vez más portátiles y al mismo tiempo más robustas, con acceso a la información de forma segura y al mismo tiempo

disponible en cualquier lugar, por ende, se ha optado por implementarlos en equipos móviles, celulares, celulares inteligentes, tabletas, etc.

Las aplicaciones para dispositivos móviles sencillamente son las que fueron creadas para adaptarse y ejecutarse en dispositivos del mismo tipo.

Lo cual implica que pueda acceder a dicha aplicación y a la información que esta implica desde cualquier parte del mundo.

Y aprovechando que los dispositivos móviles se han desarrollado considerablemente en los últimos años, obteniendo características impresionantes tanto de almacenamiento, velocidad de procesamiento y posibilidades de acceso a la internet y portabilidad, así como el diseño cada vez más compactos en tanto a hardware de los dispositivos. Esto nos abre a un mundo de servicios disponibles, entre los cuales están: acceso a mapas digitales, búsqueda de información, trasmisión multimedia, mensajería instantánea y correo electrónico, juegos en línea, etc.

Estos dispositivos han sido creados conteniendo diferentes plataformas y sistemas operativos los cuales son más pequeños y simplistas que de los de un ordenador de escritorio o laptop (Saúl Montiel Almeida, 2015).

Está considerado en existencia dos tipos de aplicaciones móviles:

- **Aplicaciones de tipo Nativo:** Estas aplicaciones son desarrollados para un tipo de sistema o plataforma de dispositivo por el motivo que contiene ciertos parámetros para un proceso de instalación exclusivo para el sistema operativo instalado en el dispositivo móvil, Adicional a esto, las aplicaciones pueden interactuar con los componentes físicos del equipo tanto como sensores GPS, Wi-Fi, receptor de

datos móviles, etc. Así mismo como varios servicios propios del dispositivo, ya sea el servicio de SMS, llamadas telefónicas, correo electrónico, etc.

- **Aplicaciones distribuidas Web:** Son aplicaciones las cuales están alojadas en servidores externos y por lo tanto deben ser ejecutadas remotamente, en la mayoría de los casos se lo realiza mediante un navegador web, una plataforma de entorno de ejecución o una aplicación distribuida previamente instalada, el último caso también entraría en el tipo de aplicación nativa. No obstante, en esta clasificación de aplicaciones no pueden tener acceso a los propios servicios o algunos componentes de hardware del equipo.

2.4.1. Desarrollo de Aplicaciones Móviles

En el mundo digital existe una inmensa variedad de tipos de dispositivos móviles y así como los sistemas y plataformas que los respaldan y gestionan por lo que desarrollar en dispositivos móviles implica tomar en cuenta las siguientes consideraciones:

- Considerar los múltiples saltos de red al acceder a recursos, información o la internet como tal por medio de los diferentes NIC[31] que pueda contener el dispositivo móvil.

- Muchos sitios a los que puede llegar a operar el dispositivo pueden ser aislados totalmente de acceso a internet o redes de comunicación.

- El hardware de un dispositivo móvil puede llegar a tener grandes variaciones en cuanto al rendimiento.

[31] **NIC:** Centro de Información de Red, dispositivo encargado de asignar direcciones IP.

- Considerar los varios sistemas operativos que existen para los dispositivos móviles, es recomendable el desarrollo para diferentes plataformas o en un entorno de ejecución multiplataforma.

- Dentro de la manufacturación de los dispositivos móviles se debe tomar en cuenta que estos son creados con distintas resoluciones de pantalla las cuales pueden afectar el funcionamiento de la aplicación si esta no se ha desarrollado de manera responsiva.

2.4.2. Requerimientos de las Aplicaciones Móviles

- Capacidad de interacción con aplicaciones en otros dispositivos, y aplicaciones de terceros.

- Interacción con el hardware del dispositivo móvil, sus respectivos NIC y sensores.

- Todas las aplicaciones deben tener estándares y técnicas de desarrollo para seguridad de la misma, solicitudes de servicios encriptados, así como código software embebido.

- Controlar la interacción con el hardware antes mencionado, pues estas operaciones requieren de batería del dispositivo móvil y este requiere que sea un consumo optimizado.

2.4.3. Contexto y uso de las aplicaciones Móviles

El uso que los usuarios finales puedan dar a las aplicaciones dependerán muchas veces de los objetivos por los que fueron desarrollados originalmente, tendrán diferentes funciones en cuanto a las tareas que puedan desempeñar, esto factores pueden depender inclusive de los entornos físicos y hasta sociales. Y todos estos parámetros pueden ser afectados por el contexto móvil y el modo de usar una aplicación por parte del usuario final.

En el caso que se necesite identificar el contexto o medir la usabilidad y sus diferentes objetivos lo cual puede ser necesario para identificar el perfil del usuario final o mercado a la cual va a estar dedicada la aplicación, es necesario analizar el escenario y entorno que se va a ejecutar la aplicación (Juan Gabriel Enriquez, Isabel Sandra Casas, 2013).

"En ese entorno real la conectividad (ancho de banda) puede ir cambiando según el lugar donde se encuentre el usuario, afectando el uso de la aplicación (Juan Gabriel Enriquez, Isabel Sandra Casas, 2013).

En los entornos en donde los recursos externos pueden tener variantes considerables como se expuso en la anterior cita es necesario analizar algunos factores:

- **Ambiente real de ejecución:** Es toda la información relevante a los usuarios que rodea al usuario final, el cual es el que utiliza la aplicación. A este factor se le puede incluir el lugar geográfico, información relevante de los usuarios que le rodean así mismo como objetos inertes, como lugares, direcciones, etc., que pues estos elementos podrían robar atención del usuario al uso de la aplicación.

- **Acceso y Conexiones:** Tanto el acceso a internet como la intensidad y velocidad de la conectividad ya sea móvil como fijo tendrá variaciones dependiendo la ubicación desde donde el dispositivo móvil esté operando, inclusive este puede afectar de sobre manera en momento de descarga masiva de datos, o trasmisiones multimedia (audio y video). Además de estos factores se debe tomar en cuenta la forma y constancia de movilidad del usuario, medios de transporte, frecuencia, etc.

- **Capacidad de procesamiento:** *"El poder computacional y la capacidad de memoria de los dispositivos móviles son reducidos con respecto a dispositivos considerados de escritorio"* (Juan Gabriel Enriquez, Isabel Sandra Casas, 2013).

- **Pantallas pequeñas:** Los dispositivos móviles por su propio nombre lo indica tienen características portables, por ende, pueden estar diseñados con pantalla muy pequeñas para las funcionalidades requeridas por ciertas aplicaciones.

- **Resoluciones Heterogéneas:** Como bien el tamaño de las pantallas puede no ser adecuado como se ha mencionado anteriormente, estás pueden utilizar una resolución de pantalla del mismo modo, afectando de este modo el uso de la aplicación y su experiencia hacia el usuario.

- **Modo de Ingreso y entrada de datos:** Nuevamente, el diseño portable del dispositivo móvil puede no ser adecuado, tomando en cuenta que se puede requerir varios componentes del dispositivo tales como el área de la pantalla táctil, teclado virtual, entre otros, a tal punto que puede no llegar a ser tan asistencial o amistoso para el usuario final.

2.4.4. Sistema Operativo Android

Android es una plataforma o sistema operativo dedicado para teléfonos celulares inteligentes, aunque se puede ejecutarlos en tabletas electrónicas, ordenadores portátiles y de mesa, entre otros dispositivos electrónicos, ya que su núcleo está basado en Linux.

"Android permite programar en un entorno de trabajo (Framework) de Java, aplicaciones sobre una máquina virtual Dalvik[32] (una variación de la máquina de Java, con compilación en tiempo de ejecución)." (Juan Gabriel Enriquez, Isabel Sandra Casas, 2013).

Adicionalmente, es posible el desarrollo de nuevas aplicaciones, franjas de notificaciones (widgets)[33], inclusive poder modificar el aspecto o funcionalidades del sistema operativo como tal.

Android fue creado por la compañía con el mismo nombre, en el año del 2005 fue adquirida por Google, no obstante, fue reconocido por los usuarios sino hasta el 2008.

Como se ha mencionado anteriormente el Kernel[34] del sistema Android, está basado en Linux, y por ende, tiene acceso a los diferentes componentes de hardware y librerías que estos pueden implicar para gestionarlos adecuadamente, existe una jerarquía en capas que representa la arquitectura del sistema operativo mostrada en la Ilustración II-6.

[32] **Dalvik:** Máquina virtual para entorno de ejecución de aplicaciones Android.
[33] **Widget:** Diminuta y sencilla aplicación la cual puede estar reflejada en un conjunto de Ficheros, es lanzado por un motor del mismo tipo.
[34] **Kernel:** Raíz o núcleo de un determinado software o sistema operativo.

2.4.5. Infraestructura por capas del Sistema Operativo Android

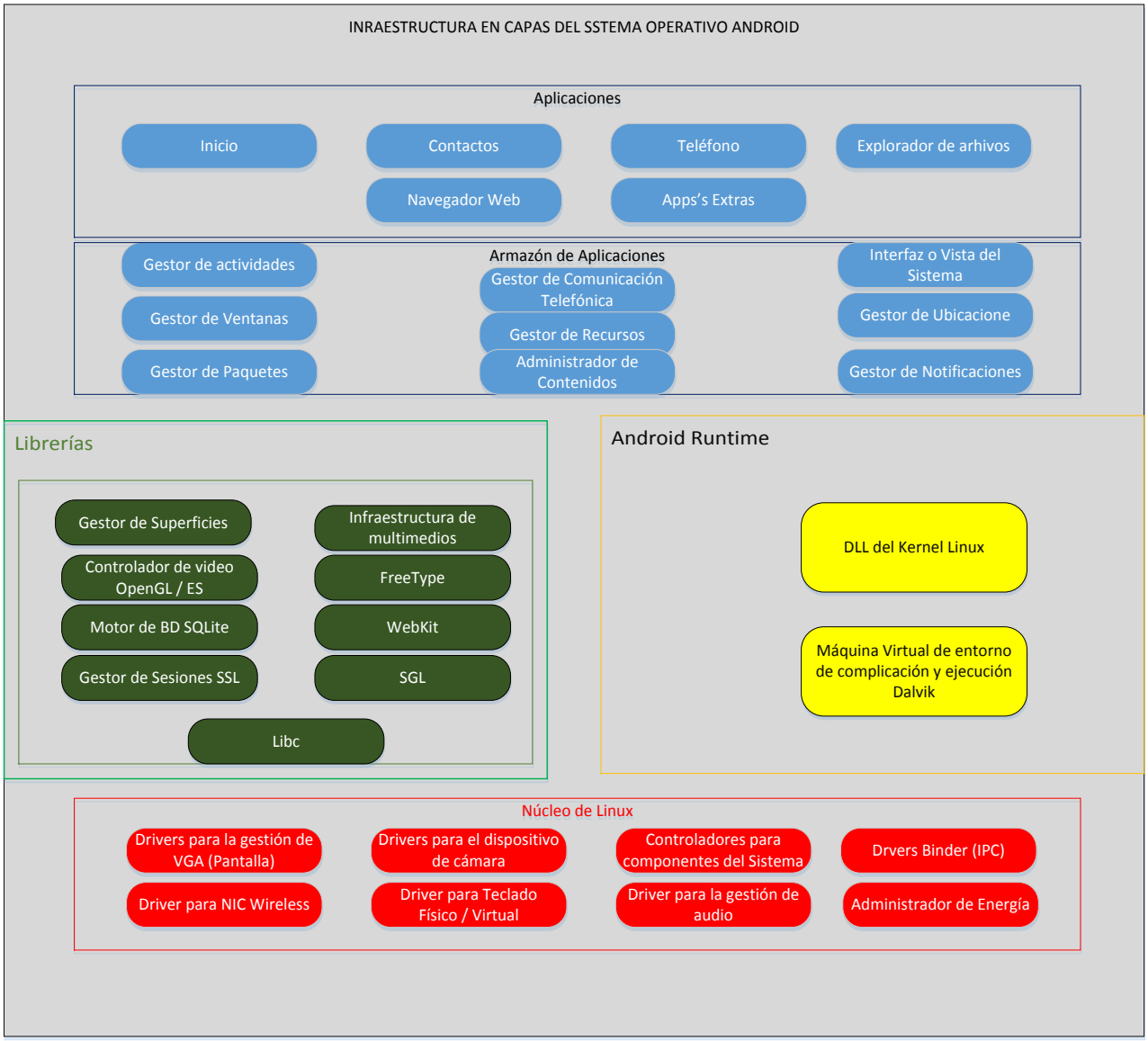


Ilustración II-6: Arquitectura por capas del Sistema Operativo Android

Elaborado por: Diego Ponce – Juan Prado

Basado de: (Basterra, Bertea, Borello, Castillo, Venturi, 2016)

2.4.6. Versiones del Sistema Operativo Android

Tabla II-1: Versiones del Sistema Operativo Android hasta junio del 2016

No.	Nombre de Versión	Fecha de Lanzamiento	No. de Versión del SO.
1	Apple Pie	23/9/2008	1.0
2	Banana Bread	9/2/2009	1.1
3	Cupcake	27/4/2009	1.5
4	Donut	15/9/2009	1.6
5	Eclair	26/10/2009	2.0-2.1
6	Froyo	20/5/2010	2.2-2.2.3
7	Gingerbread	6/12/2010	2.3-2.3.7
8	Honeycomb	22/2/2011	3.0-3.2.6
9	Ice Cream Sandwich	18/10/2011	4.0-4.0.4
10	Jelly Bean	9/7/2012	4.1-4.3.1
11	KitKat	31/10/2013	4.4-4.4.4/ 4.4W-4.4W.2
12	Lollipop	12/11/2014	5.0-5.1.1
13	Marshmallow	5/10/2015	6.0-6.0.1

14	Nougat	15/6/2016	7.0

Elaborado por: Diego Ponce – Juan Prado

Basado de: (Orero, 2016)

2.4.7. Kit de desarrollo de Software de Android (SDK)

El SDK[35] para Android es una plataforma de desarrollo en el cual está contenida todas las herramientas y complementos para el diseño y la creación de una aplicación en dicho sistema operativo. Este SDK es multiplataforma, por ende, es compatible con sistemas operativos ya sea núcleo Windows, Linux, etc.

2.4.7.1. Componentes del SDK de Android

- **Puente de Depuración de Android (Android Debug Bridge- adb):** Este componente tiene la responsabilidad de crear y mantener estable un enlace desde el IDE[36] de desarrollo hacia el dispositivo android, o un emulador del mismo tipo. La herramienta transfiere el código hecho aplicación para poder ejecutarlo en entornos reales, ya sea un el dispositivo físico o el emulador.

- **Ensamblador Dalvik (Dalvik Cross-Assembler - dx):** La presente herramienta tiene la capacidad de agrupar todas las clases Java que conforma la aplicación y fusionarlas, una vez hecho este proceso lo convierte en un archivo binario con una extensión *.dex, los cuales son compatibles con la máquina virtual

[35] **SDK:** Serie de herramientas para el desarrollo de Software en un lenguaje o plataforma determinada.
[36] **IDE:** Entorno para el desarrollo de Software Integrado.

Dalvik que tiene por defecto el sistema operativo de Android. (Sylvain Hébuterne, Sébastien Pérochon, 2014)

- **Servicio de Monitoreo de Depuración Dalvik (Dalvik Debug Monitor Service – ddms):** Este componente es utilizado para la depuración de proyectos de desarrollo de aplicaciones en Android y contiene las siguientes características:

  - Permite visualizar los hilos[37], procesos o demonios que se realiza mientras la aplicación se ejecuta en primer plano.

  - Visualiza el registro de eventos que pueda producir la aplicación.

  - Tiene la capacidad de realizar capturas de pantalla de la aplicación en ejecución.

  - Lista los procesos en ejecución provenientes del dispositivo como tal.

  - Simulación para interactuar con el servicio de mensajes y llamadas del dispositivo móvil.

  - Simulación para el uso de HW[38] del GPS y servicio de ubicación del dispositivo móvil.

[37] **Hilo:** Sub proceso o proceso secundario el cual genera un servicio de cierta aplicación y se ejecuta comúnmente en segundo plano.
[38] **HW:** Acrónimo de Hardware, parte física de un ordenador.

- Contiene un administrador de versiones del sistema Android, así como para el uso de sus API's y su respectiva documentación.

2.4.8. Entorno de Desarrollo Integrado (IDE) – Android Studio

En el año 2003 durante la conferencia de Google I/O la compañía con el mismo nombre lanzó al público orientado al desarrollo de aplicaciones móviles un IDE exclusivo para el sistema operativo Android, se convirtió en una gran ventaja al ya no tener la dependencia del IDE Eclipse para crear aplicaciones en dicho sistema operativo.

Android Studio está basado en el IDE nativo IntelliJ IDEA[39], el mismo que por su licencia de uso libre, está abierto para todo tipo de público, El IDE ya contiene dentro de su paquete de instalación la plataforma de desarrollo SDK. (Sylvain Hébuterne, Sébastien Pérochon, 2014)

Este IDE de desarrollo ofrece entre sus principales características:

- Construcción y compilación de varios archivos ejecutables con extensión *.apk para varias versiones del sistema operativo nativo.

- Auto generación de código para ayudar al diseño de la aplicación móvil, y módulos que pueden ser comunes en el mismo.

- Su sistema está basado en Gradle Flexible[40].

[39] **IntelliJ IDEA:** Entorno para desarrollo IDE desarrollado por la empresa JetBrains.
[40] **Gradle Flexible:** Sistema de automatización para la construcción de proyectos Software.

- Tiene integrado en el propio IDE el servicio de nube de Google (Cloud Plataform Google), por este motivo es fácil implementarlo y hacer uso de sus diferentes API's y servicios.

2.4.8.1. *Componentes del Desarrollo de Proyectos en Android Studio*

- **Entorno de la Aplicación:** Es toda la interfaz para el desarrollo de la aplicación el cual proporciona una gran variedad de herramientas, componentes y servicios para lograr este objetivo. Está diseñado para simplificar el desarrollo de código común e innecesario además de asistente para la depuración de errores. Entre las características y servicios que ofrece el entorno están (Sylvain Hébuterne, Sébastien Pérochon, 2014):

    - **Vistas:** Variedad de Perspectivas para la gestión de clases y recursos de la aplicación

    - **Administrador de recursos:** Gestiona y proporciona acceso a archivos o componentes ajenos a la edición de su código fuente.

    - **Administrador de Actividades:** Gestiona el flujo de la aplicación o también conocido como el ciclo de vida de la misma añadiéndoles parámetros de navegación entre ellos.

- **Administrador de Notificaciones:** Está encargada de gestionar alertas o notificaciones en la barra de estado de la misma.

- **Proveedor de contenidos:** Módulo que brinda acceso a aplicaciones externas o que pueden proporcionar información de terceros como la agenda de contactos, servicio de mensajería, NIC's del dispositivo, etc.

- **Vista de Proyectos:** Perspectiva del entorno IDE que permite visualizar y tener acceso a la edición de archivos que son relevantes para el desarrollo del proyecto tanto como las clases de Java como sus paquetes, el manifiesto del proyecto y los recursos que utilizará la aplicación en su base Gradle.

- **Tiempo de Ejecución de Android:** Tomando en cuenta que Android empezó a ejecutarse en dispositivos con memoria muy limitada y velocidades realmente bajas de procesamiento, por tal motivo no fue posible lanzar una máquina virtual de Java, se llega al caso de crear una nueva máquina virtual basada en la anteriormente mencionada llamada Dalvik, la cual pudiese acoplarse mejor a características sumamente bajas de hardware. Dalvik compila las aplicaciones en su propia extensión (.dex) y utiliza un hilo exclusivo de Linux para cada una de ellas, con el fin de optimizar memoria, por lo cual ese funcionamiento también está basado en registros (Sylvain Hébuterne, Sébastien Pérochon, 2014).

2.4.8.2. Librerías incluyentes en Android Studio

Varios componentes existentes en Android requieren de librerías las cuales están compiladas o creadas en C o C++, inclusive en código propio del micro procesador del dispositivo. Entre las más destacadas están:

- **Librerías de Sistema C:** Dedicada para los dispositivos o HW que está en constante interacción con el núcleo Linux.

- **Plataforma de medios:** Librería que proporciona servicios para soporte de codificadores multimedia e imágenes.

- **Administrador de Superficies:** Administra sistemas de gráficos ya sea estos en dimensiones 2D o 3D.

- **WebKit:** Librería para interactuar con navegador web y su respectiva vista.

- **Librería de Gráficos Escalable SGL:** Motor gestor de gráficos en 2D.

- **Librerías de gestión 3D:** Estas librerías están basadas en la plataforma gráfica OPENGL la cual utiliza las API's propia de las mismas, tienen la capacidad el hardware de aceleración en 3D sea el caso, o el SW[41] de proyección para el mismo tipo de funcionamiento.

- **Fuente FreeType:** Fuentes basado en mapa de bits los cuales están caracterizados por basarse en un rende rizado vectorial.

[41] **SW:** Acrónimo para el termino de Software, contenido digital o virtual de un ordenador.

- **Motor SQLite**: Es un motor de base de datos dedicado para aplicaciones móviles.

- **Gestor de Sesión SSL:** Librería con servicios de encriptación para Sockets[42] de Seguridad.

2.4.8.3. Componentes importantes que conforman un proyecto de desarrollo móvil Android

La arquitectura de un proyecto en desarrollo de una aplicación en Android, sea el caso que sea en lenguaje Java y para la máquina Dalvik, está conformada por tres componentes: el descriptor, o fichero de descripción de la aplicación (AndroidManifest.xml), los ficheros contenedores de código fuente, y los recursos adicionales, los cuales se representa en el siguiente árbol de directorio:

- **Carpeta "Src":** Dentro de este directorio se encuentra todas las clases Java con su respectivo código fuente.

- **Carpeta "Gen":** Aquí se encuentra ficheros con código auto generado, ya sea por la propia plataforma SDK o a su vez el IDE de Android, es recomendable no modificar el código de estos ficheros si no es necesario.

- **Fichero "R.java":** Se encuentra dentro de la carpeta "Gen", contiene código que permite asociar todos los componentes recursos de la aplicación con un

[42] **Socket:** Puerto virtual o físico de un ordenador o sistema operativo, el cual sirve para conectar un dispositivo físico o comunicarse mediante una red respectivamente.

identificador predeterminado, logrando así un enlace de consumo de recursos desde el lenguaje de Java.

- **Android x.x:** Contiene código auto generado JAR[43], y los componentes API's del sistema Android dependiendo de su versión.
- **Carpeta "Assets":** Es un directorio anexo del proyecto de la aplicación ya que puede ser para almacenar diferentes recursos de datos como ficheros, archivos JAR de terceros, fuentes de texto, entre otros.
- **Carpeta "Res":** Contiene todos componentes y recursos que estarán vinculados directamente a la aplicación por medio de identificadores dentro del código en las clases Java.
- **Carpeta "Drawable":** Directorio con contenido multimedio, como imágenes y descriptores de los mismos.
- **Carpeta "Layout":** Está conformado por ficheros de extensión *.XML, los mismos que son los archivos con código de diseño de interfaces de las diferentes pantallas, fragmentos o vistas que pueda contener la aplicación.
- **Carpeta "Menú":** Contiene ficheros similares de la carpeta Layout[44], no obstante, son vinculados al menú de la aplicación, en el caso que lo tenga.

[43] **JAR:** Representación de la extensión de archivos tipo Java (Java Archive).
[44] **Layout:** Croquis o diseño preliminares de la interfaz de una aplicación determinada.

- **Carpeta "Values":** En esta carpeta está contenida archivos de tipo XML[45], cuyo código representa a un arreglo de datos tipo cadena de caracteres (String)[46] que servirán para hacer referencia a dichos valores en el código Fuente de Java, de esta manera solo son invocado con sus respectivos descriptores. Un ejemplo de aplicación para este tipo de archivos puede ser las diferentes traducciones de idiomas de una aplicación.
- **Carpeta "Anim":** Este directorio contiene archivos XML símiles al de las anteriores carpetas mencionadas.

- **Ficheros "Row":** Archivos de terceros que no están en formato de páginas XML.

- **Carpeta "Doc":** Se encuentra toda la documentación perteneciente al proyecto de la aplicación.

- **Fichero "AndroidManifest.xml":** Este es un fichero de vital importancia ya que contiene toda la información básica del proyecto, así como las actividades que utilizará y el flujo de navegación de la misma, así mismo tiene definido todos los servicios y componentes que necesitará utilizar desde el HW del dispositivo, adicionalmente también define todas las API's que ocupa dentro de la aplicación. Esta información debe ser especificada ya que antes de la instalación siempre se pide dicha verificación desde el sistema operativo Android.

[45] **XML:** Acrónimo para el Lenguaje de marcación de Hiper texto Extensible.
[46] **String:** Tipo de dato que representa una cadena de caracteres dinámica o predeterminada.

- **Fichero "Default.propiertes":** Este archivo es auto generado por la plataforma de desarrollo, este es usado para validar la versión de API de Google para Android y características de monitorización en el terminal del adb.

2.4.8.4. Componentes que conforman una Aplicación de Android.

Una aplicación ya completa de Android puede componerse de uno o varios componentes dependiendo de su complejidad y funcionalidad, desde un módulo de actividad hasta su propia pantalla de presentación (Splash Screen) y sus diferentes pantallas de actividades y fragmentos.

Una aplicación móvil perteneciente al sistema Android puede conformarse de los siguientes componentes:

- **Vista de la Aplicación – View:** Estos componentes están definidos como todo lo que conforma la interfaz y el usuario final puede apreciarlo a primera vista, ya sea el entorno de la aplicación, botones, caja de textos, listas, casilleros de verificación, etc. Todos estos componentes u objetos son tomados de la herencia de la clase con el nombre View y por supuesto pueden ser invocados en su respectivo código Java, no obstante, comúnmente se lo define desde el paquete "Layout" mediante un recurso XML como se definió anteriormente.

- **Paquete de Diseño – Layout:** Este paquete se define como la agrupación de dos o más vistas de forma que contenga una posición definida, al igual que el caso

anterior estos pueden ser declarados o invocados desde código Java, pero usualmente se lo hace desde un recurso XML.

- **Actividad de la Aplicación – Activity:** Todos los componentes visuales que componen una aplicación de Android, tanto como sus pantallas, objetos, fragmentos, etc., en conjunto está definido como la Actividad de la aplicación, la cual tiene como responsabilidad principal la generación total de la interfaz para el usuario final. Cada aplicación puede tener más de una actividad, estas serán independientes entre sí, no obstante, tendrán un mismo enfoque y meta dentro de la aplicación móvil.

- **Servicio de la Aplicación – Service:** Los servicios básicamente son demonios que están trabajando en segundo plano (background) de la aplicación en el sistema operativo, no obstante, está en constante interacción con el usuario. Los servicios pueden estar clasificados por dos tipos de los mismos: Los servicios locales, propios de Android que pueden ser ejecutados directamente por las aplicaciones inclusive por una terminal de comandos, y servicios remotos los cuales primero son invocados desde un servidor de aplicaciones o desde otro terminal de comandos en otro dispositivo.

- **Administrador para la escucha de anuncios (Broadcast Receiver):** Son componentes con eventos de escucha (Listeners) para notificaciones generales o que aplican a toda aplicación (Broadcast)[47]. Estos pueden ser causados por aplicaciones o servicios nativas del dispositivo móvil, como el de llamadas, notificaciones de consumo o nivel de batería del dispositivo. Estos no tienen una vista o interfaz de

[47] **BroadCast:** Método de comunicación masiva de paquetes por medio de las redes de información digital.

usuario como tal, no obstante, pueden iniciar una actividad para tener interacción con el mismo.

- **Intención de una aplicación móvil – Intent:** La intención es la invocación de una acción o evento de una aplicación Android la cual se ejecutará dependiendo la voluntad del usuario, dentro de esta puede involucrar las siguientes acciones:

    - Lanzamiento o invocación de un servicio.
    - Lanzamiento o invocación de una actividad dentro de la aplicación.
    - Lanzamiento de notificaciones tipo Broadcast dentro de la aplicación.
    - Interacción con servicio ajeno o tercero.

Los componentes, servicios o demonios que sean generados o lanzados por la aplicación pueden ser tanto externos como locales al igual que puede haber una comunicación entre sí de estos y habrá casos excepcionales en donde una intención sea generada o lanzada nativamente por el sistema operativo para alguna notificación o servicio requerido ya sea por la aplicación desarrollada o acción del usuario final.

2.5. BASE DE DATOS

En la actualidad la información para una empresa es muy importante y esta puede implicar inclusive la estabilidad y éxito de la misma ya sea como datos de cartera de clientes, datos financieros, estadísticas de ventas y mercado, etc. Por este motivo se ha desarrollado durante la historia de la informática diversas técnicas para la gestión de datos (Abraham Silberschatz, Herny F. Korth, S. Sudarshan, 2002).

El margen para la aplicación de la administración de la información es tan amplio como la puede tener toda lógica de negocio, entre las más destacables podemos denotar:

- Entidades Financieras y Bancarias.
- Entidades escolares, de educación superior, y de investigación.
- Sistemas de telecomunicaciones.
- Aeropuertos y entidades de viajes.
- Departamentos de venta y marketing.
- Entidades industriales y de producción.
- Departamentos de recursos humanos.

Tomando en cuenta la era de la digitalización una forma de mantener la información estable y almacenada seguramente es en un ordenador en respectivos archivos dentro de su disco duro por medio de un motor de base de datos SGBD.

"Un sistema gestor de base de datos (SGBD) consiste en una colección de datos interrelacionados y un conjunto de programas para acceder a dichos datos. La colección de datos normalmente denominada base de datos, contiene información relevante para una empresa." (Abraham Silberschatz, Herny F. Korth, S. Sudarshan, 2002).

Adicionalmente, un motor de BD[48] debe tener la capacidad de brindar una infraestructura interactiva con el usuario administrador, para que pueda manipular los datos y una serie de programas para lograr el último objetivo mencionado.

"La gestión de base de datos implica tanto la definición de estructuras para almacenar la información como la provisión de mecanismos para la manipulación de la información. Además, los sistemas de base de datos deben proporcionar la fiabilidad de información

[48] **BD:** Acrónimo para el término de Base de Datos.

almacenada, a pesar de las caídas del sistema y los intentos de acceso sin la autorización. Si los datos van a ser compartidos entre diversos usuarios, el sistema debe evitar posibles resultados anónimos" (Abraham Silberschatz, Herny F. Korth, S. Sudarshan, 2002).

2.5.1. Posibles problemáticas dentro de un Sistema Gestor de Base de Datos y su respectiva administración.

Incluyendo la anterior problemática, en la administración de la base de datos para una empresa u organización se pueden generar los siguientes inconvenientes importantes a tomar en cuenta:

- **Redundancia o inconsistencia de la información en un BD:** Puede existir inconvenientes al registrar información en una base de datos y que esta pueda estar duplicada o así mismo encontrarse con duplas anteriores en la misma, ya que esto pueda ser causado ya sea por un error del usuario final (conjuntamente con una aplicación no validada), o por aplicaciones creadas en diferentes plataformas o lenguajes de programación que estén consumiendo del mismo recurso. Esto en el proceso de información financiera puede requerir costos de reparación, adicional que puede crear una inconsistencia de datos, ya que los datos que deben estar estables en un registro no están actualizados en otros a lo que tengan relaciones directas (Abraham Silberschatz, Herny F. Korth, S. Sudarshan, 2002).

- **Dificultad en el acceso a datos:** La información puede llegar a ser relacionada, dependiente e inclusive anidada, a esto se añade seguridades para que los usuarios puedan acceder a información exclusiva a sus necesidades y autorizaciones,

no obstante, el proceso puede ser ineficiente al momento de querer ser accedida, para esto se necesitaría ya sea obtenerlos de manera manual o física, o desarrollar programas que permitan hacer este proceso por medio de una interfaz amigable, estos requerimientos se los hace comúnmente a los departamentos de IT[49], o de desarrollo de software.

- **Aislamiento de la información en la DB:** La información puede estar dispersa, fragmentada, en diferentes archivos de diferentes tipos o extensiones, la dificultad se incrementa considerablemente al querer desarrollar nuevas aplicaciones para administrar dicha información.

- **Integridad de la información:** La información que contiene una base de datos debe estar relacionada con la información básica de la empresa al igual que con su lógica de negocio. Adicional a esto, los datos almacenados deben respetar las políticas, estándares y restricciones de la misma, unidades de medida, cantidades y valores que bien tienen que cumplir límites y rangos para que puedan tener aprobación de manipulación son algunos ejemplo de restricciones que puede tener una empresa y su base de datos, esta problemática es un tanto compleja ya que el desarrollo de aplicaciones debe estar acorde con las diversas resoluciones y condiciones en las políticas empresariales, y al tanto de sus respectivas actualizaciones.

- **Inconvenientes de Atomicidad en la información:** Como cualquier dispositivo electrónico, ordenador o hardware del mismo, puede estar expuesto a

[49] **IT:** Acrónimo para el término de Tecnología de la Información o Informática.

fallos, apagones y reinicios inesperados o no planeados por el administrador o usuario encargado. Debido a esto, es importante ya sea que los motores de BD o las aplicaciones que gestionan y manipulan estos datos tengan embebidos software de seguridad y validación de datos, para que no sean borrados y haya inconsistencia en alguno de los dispositivos que tenga acceso.

- **Inconvenientes en sesiones constantes con la información:** Se puede lograr de igual manera una inconsistencia de datos en sesiones simultáneas de la manipulación de la información, específicamente de la actualización recurrente de un mismo dato ya sea por diferentes usuarios o aplicaciones. El motor que gestiona los datos tiene la responsabilidad de monitorear y supervisar los diferentes accesos a un mismo dato y coordinar las diferentes peticiones que puede haber entre las aplicaciones exteriores.

- **Inconvenientes con la seguridad de la información:** No todos los usuarios pueden tener acceso a toda la información de una base de datos, y esta problemática debe estar tomada en cuenta tanto como el gestor del BD como la aplicación que la consume. Adicionalmente se debe pre configurar los respectivos perfiles de acceso para cada usuario y así solo disponerle la información que verdaderamente necesita para cumplir su trabajo.

2.5.2. Objetivo del Sistema Gestor de Base de Datos

Los sistemas de base de datos pueden tener características o habilidades un tanto ocultas que en conjunto operan para cumplir con el objetivo de almacenar y gestionar la información, así como es la abstracción de datos.

2.5.3. Abstracción de Datos en Información

Como bien se ha apreciado en los anteriores postulados referente al tema de base de datos puede existir una gran complejidad al intentar al tratar de consultar y manipular la información de manera eficiente, al igual que muchos usuarios se les dificulta el uso de esta tecnología, incluso existe gran complejidad de los usuarios al poder usar ordenadores de manera adecuada, por esto, los desarrolladores siempre intentan reducir de manera considerable, o casi nula esta complejidad por medio de determinados niveles de abstracción, para así lo único transparente sea la información o los datos requeridos (Abraham Silberschatz, Herny F. Korth, S. Sudarshan, 2002).

Los niveles de abstracción están clasificados de la siguiente manera:

- **1er. Nivel de Abstracción – Físico:** Se refleja la manera real del almacenamiento de datos, es decir, a bajo nivel y se describe su infraestructura a detalle.

- **2do. Nivel de Abstracción – Lógico:** En este nivel se define que datos serán almacenados en la base, incluyendo sus tablas y relaciones, agrupándose en un número simple de estructuras dentro de la base. Este nivel no es administrable ya que

no tiene alguna responsabilidad, y otros módulos o aplicaciones exteriores realizarán dicha gestión.

- **3er. Nivel de Abstracción – Vistas:** El nivel más alto de la abstracción se enfoca en solo a partes exclusivas y necesarias de la base de datos, dependiendo del requerimiento del usuario final lo cual hace que la información sea clasificada en vistas, pues no todos los usuarios necesitarán o tendrán autoridad de acceso a toda la información de la base de datos.

2.5.4. Modelo de Datos e Información

Dentro del proceso de escritura y consulta de datos se puede referir a ciertas herramientas, reglas y condiciones o delimitantes las cuales conformar y definen el modelo de datos, estos están clasificados de la siguiente manera:

- **Modelo E/R (Entidad – Relación):** Es una abstracción de la realidad o la lógica del negocio sea el caso, cada objeto que está conformado o declarado desde el mismo se define como entidades, una entidad es un objeto extraído o representante del mundo real independiente entre sí de otros objetos o entidades. Al declarar la entidad en una base de datos se lo debe hacer conjuntamente con una serie de atributos que serán componentes descriptivos necesarios para saber que contiene o qué tipo de información almacenará dicha entidad.

  Cada entidad tendrá un identificador único definida como la clave primaria. Entre los componentes de una entidad también se encuentra la relación, cuyo objetivo es generar una asociación entre una o varias entidades. Para representar un modelo E/R es necesario realizar un diagrama el cual tendrá al menos los siguientes componentes:

- **Rectángulos:** Tienen la funcionalidad de representar entidades.
- **Elipses:** Este componente refleja los atributos de una entidad
- **Rombos:** Refleja un conjunto de entidades
- **Líneas:** Crean la asociación que pudiera haber entre entidades.

Un claro y sencillo ejemplo de la aplicación de este diagrama se lo puede reflejar en el diseño de una base de datos para un simple sistema de matriculación como muestra la Ilustración II-7.

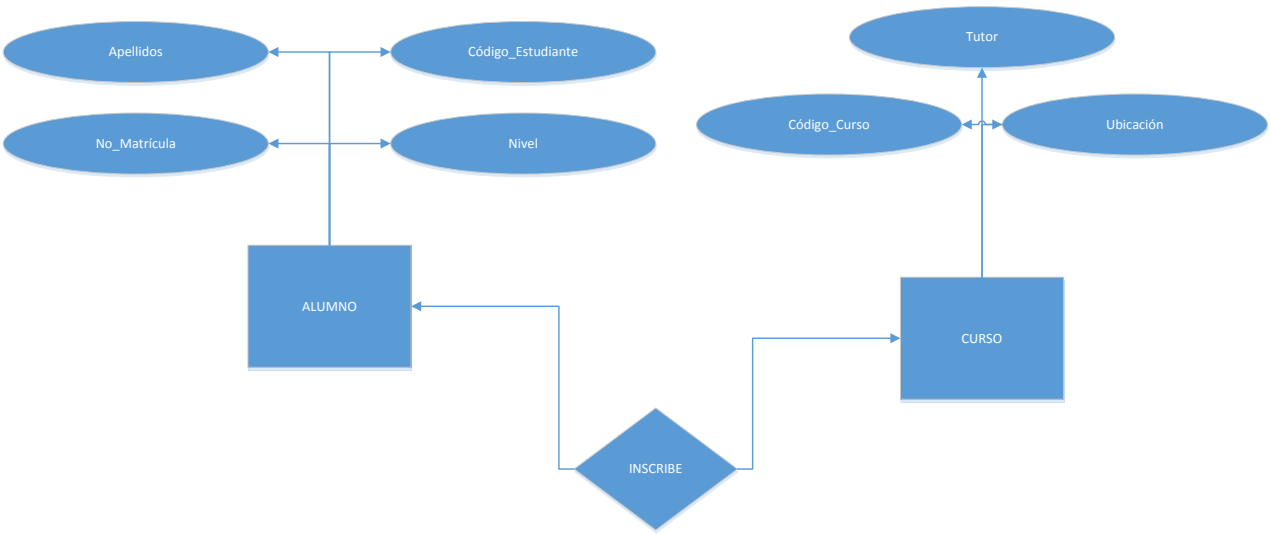


Ilustración II-7: Ejemplo del modelo de Datos E/R

Elaborado por: Diego Ponce – Juan Prado

En la Ilustración II-7 se aprecia dos entidades: "Alumno" y "Curso", cada uno con sus respectivos atributos mostrados en sus círculos adyacentes, finalmente se define la relación "Inscribe" que vincula a las dos entidades antes mencionadas.

El modelo E/R adicionalmente implica varias políticas o restricciones al momento de diseñar los diagramas de BD.

Una de las restricciones más destacadas es de la cardinalidad, ya que en una relación cualquiera de las dos entidades puede necesitar un número determinado de la anterior mencionada, es decir, varias entidades de "Estudiante" puede estar involucradas en una sola entidad de "Curso" sea el caso.

- **Modelo de datos Relacional:** Este modelo se usa para el diseño de base de datos por medio de un conjunto de tablas en donde están representados los mismos, cada una de las tablas puede estar conformada por columnas las cuales pueden ser definidas como los atributos en el caso del modelo E/R, y claramente cada uno de ellos tendrá su identificador único y principal.

Se presenta el mismo ejemplo del modelo E/R ahora en el caso del modelo relacional en las Tablas II-2, II-3 y II-4.

Tabla "Estudiante"

Tabla II-2: Tabla de ejemplo "Estudiantes" para el modelo de Datos Relacional

cod_estudiante	nombre_estudiante	apellido_estudiante	nivel_estudiante	no_matricula
1	Juan	Carranza	3	122
2	Josue	Lopez	5	12324
3	Christian	Vega	2	14654
4	Ana	Quintanilla	4	3256
5	Pamela	Ducal	5	4356
6	Oscar	Calvache	6	434

Elaborado por: Diego Ponce – Juan Prado

Tabla "Curso"

Tabla II-3: Tabla de ejemplo "Curso" para el modelo de Datos Relacional

cod_curso	tutor_curso	ubicacion_curso
1	Carlos Salazar	308
2	Fabian de la Cruz	201
3	Damían Nicolalde	406
4	Gustavo Chafla	511
5	Oswaldo Espinoza	311
6	Édison Mora	304

Elaborado por: Diego Ponce – Juan Prado

Tabla "Inscribe"

Tabla II-4: Tabla relacional de ejemplo "Inscribe" para el modelo de Datos Relacional

cod_estudiante	cod_curso
2	2
3	3
2	4
2	2
1	1
5	2

Elaborado por: Diego Ponce – Juan Prado

Como se puede apreciar en las tablas anteriores en la primera tabla se obtiene datos de solamente de los estudiantes con sus respectivos atributos que en este caso sería cada columna, se toma en cuenta que cada estudiante tiene un código identificador que se ha definido en la primera columna "cod_estudiante", mientras en la tabla de "curso" está definido la información de cada aula donde mantendrá clases los estudiantes, esta información se vincula directamente y se lo puede observar en la tabla "Inscribe" como puede estar formada la relación.

"El modelo relacional es un ejemplo de un modelo basado en registros. Los modelos basados en registros se denominan así porque la base de datos se estructura en registros de formato fijo de varios tipos. Cada tabla contiene registros de un tipo particular. Cada tipo de registro define un número fijo de campos, o atributos. Las columnas de la tabla corresponden a los atributos del tipo de registro." (Abraham Silberschatz, Herny F. Korth, S. Sudarshan, 2002).

- **Otros modelos de base de datos:** Existen adicionalmente modelos que están en constante evaluación y actualización como las bases de datos orientadas a objetos, las cuales pueden considerarse como una adaptación o a su vez una evolución del modelo E/R con teorías de encapsulamiento, conjunto de métodos o técnicas que estarán relacionados con objetos o identidades de datos.

2.5.5. Lenguaje de Base de Datos

Si bien para especificar un esquema de base de datos se necesitará respectivamente su lenguaje para la definición de datos de la base, y adicional un lenguaje de datos para la misma.

En la aplicación tanto los lenguajes de definición como de como de manipulación de datos funcionan en un mismo entorno que conforman un lenguaje único y

constituido llamado SQL[50], el cual es ampliamente usado en el mundo de las bases de datos (Abraham Silberschatz, Herny F. Korth, S. Sudarshan, 2002).

2.5.5.1. Lenguajes para la definición de datos en una BD

"Un esquema de base de datos se especifica mediante un conjunto de definiciones expresadas mediante un lenguaje especial llamado lenguaje de definición de datos (LDD)." (Abraham Silberschatz, Herny F. Korth, S. Sudarshan, 2002).

Además de estas operaciones, es necesario actualizar una tabla cuya información está definida como el diccionario de datos, en el cual estará contenido metadatos de la información almacenada en la base de datos.

Antes de realizar un CRUD[51] dentro de la base de datos, la tabla del diccionario de datos es consultada previamente esta operación.

Los LDD son métodos e instrucciones que definen de manera detallada la implementación de la estructura de la base, esta información comúnmente no está disponible para los usuarios.

Otra de las condiciones de los LDD es regirse a la consistencia de datos dentro de la base, y tomar en cuenta la gestión de solicitudes recurrentes a un recurso en la misma (Abraham Silberschatz, Herny F. Korth, S. Sudarshan, 2002).

2.5.5.2. Lenguajes para la manipulación de datos en una BD

[50] **SQL:** Acrónimo para el sistema de Lenguaje de consultas Estructurado para base de datos.
[51] **CRUD:** Acrónimo para las actividades principales en contenidos de información digital: Crear, Leer, Actualizar y Eliminar.

Se puede definir a la manipulación de datos dentro de una base de datos con los siguientes estipulados:

- Consulta de la información ya almacenada dentro de la base de datos.

- Insertar nueva información dentro de la base de datos.

- Eliminación de registros predeterminados en la base de datos.

- Modificación de información previamente insertada dentro de la base de datos.

Un LMD (Lenguaje de Manipulación de Datos), pone a disposición del usuario administrativo o final el acceso para la modificación de los datos que están dentro de la base tomando en cuenta su respectivo de modelo en el que este está diseñado. Se toma en cuenta dos clases de LMD:

- **Lenguajes Procedimentales:** Están definidos por el usuario, el mismo que por medio de ciertos comandos, métodos y parámetros se determina cuáles son los datos deseados y la forma de obtenerlos.

- **Lenguajes Declarativos:** De igual manera son definidos por el usuario, no obstante, solo especifica los datos deseados, mas no como obtener los mismos.

El hecho de que en un LMD declarativo no se especifique la manera de obtener los datos, no significa que no sea necesario, el motor de la base tiene que hacerlo de la manera más factible y eficiente, como en el caso de SQL el cual es LMD declarativo no procedimental.

En SQL se declara las solicitudes de acceso a la información mediante consultas, las mismas que sirve para dicho objetivo, esta herramienta está definida como los lenguajes de consultas de base de datos.

Dentro de las consultas se puede vincular accesos a información más de una tabla de datos, ya sea que estén o no relacionadas entre sí (Abraham Silberschatz, Herny F. Korth, S. Sudarshan, 2002).

2.5.5.3. Lenguajes de aplicación para el acceso a la Base de Datos

Los lenguajes o aplicaciones de acceso de datos son plataformas de alto nivel para el desarrollo de programas vistas a la base de datos, estas aplicaciones están provistas de un lenguaje nativo el cual generará la vinculación, ya sea C#, C estructurado, Java, C++, Phyton, etc., para que el proceso de acceso de información sea ejecutado exitosamente desde un lenguaje anfitrión, hará falta ejecutar las sentencias de consulta SQL desde el mismo.

Se toman en cuenta dos formas de realizar este proceso:

- Utilizando plataformas y librerías adicionales a los programas de aplicación ya sea como un estándar de conectividad a base de datos abierto y que es derivado para cada lenguaje respectivamente como, por ejemplo, ODBC[52] para Microsoft, JDBC[53] en el caso de Java.

- Instalando librerías o controladores adicionales a los lenguajes de programación con el objetivo de expandir sus comandos y sintaxis de manera que se pueda incorporar LMD y de esta manera realizar consultas SQL dentro de la misma.

[52] **ODBC:** Estándar libre para el acceso a base de datos, realizada para plataformas Microsoft.
[53] **JDBC:** Estándar libre para el acceso a base de datos, exclusiva para plataformas Java.

2.5.6. Motor de Base de Datos MySQL

MySQL es conocido por ser un sistema motor para la gestión de información en una base de datos y como cada una de las extensiones de SQL en lo que respecta a motores y gestores, este tiene características que lo destacan entre los demás como principalmente la licencia de su uso y manipulación la cual es totalmente libre y está respaldada por el estándar de licencia GNU GPL[54], por lo que principalmente su uso es en sistemas Linux, a pesar de que también existen para plataformas y sistemas de Microsoft, de igual manera su código es abierto al público desarrollador para su respectiva contribución y crecimiento (Abraham Silberschatz, Herny F. Korth, S. Sudarshan, 2002).

En cuanto al soporte y asistencia del motor, tiene una documentación altamente disponible en internet, así como foros y blogs de ayuda, sin embargo, existe una asistencia con una membresía pagada.

Adicionalmente cuenta con un sistema amigable para su uso y administración, con un amplio set de herramientas que incrementan considerablemente la potencia del motor.

En un principio MySQL solo fue pensado para algunas plataformas y aplicaciones web, no obstante, este motor fue creciendo hasta ser considerado una solución factible y muy destacada al momento de la administración de datos e información.

2.6. SERVICIOS WEB

[54] **GNU/GPL:** Tipos de licencias para uso y creación de software.

Un Servicio Web (Web Service) puede estar definido por un componente que está alojado en un determinado servidor de aplicaciones, el mismo que puede soportar este tipo de función, y puede ser accedido por cualquier dispositivo que tenga acceso a este y pueda comunicarse por medio de ciertos métodos y parámetros, la vía de acceso será la misma Internet o bien sea el caso en entornos de aplicaciones distribuidas (Cliente - Servidor) de manera directa (Fernando Pech-May, Mario A. Gomez-Rodrigez, Luis A de la Cruz-Diaz, Salvador U. Lara-Jerónimo, 2010).

2.6.1. Ventajas de aplicaciones desarrolladas en Entornos Web

- Reduce de manera considerable el costo de implementación a comparación de desarrollo de aplicaciones de escritorio.

- La disponibilidad de la información es expandida a ser accedida desde cualquier parte del mundo que cuente con conexión a internet.

- El desarrollo de servicios web se ha convertido en estándar para el consumo de recursos en cuanto respecta a la información digital.

- Contribuyen a la interacción de sistemas de plataforma diferente eliminando así tiempo y costo de desarrollo en terceras tecnologías.

2.6.2. Desventajas de aplicaciones desarrolladas en Entornos Web

- Si bien el costo puede ser reducido considerablemente por el ahorro de desarrollo, puede incrementarse por el hecho de implementar infraestructura o a su vez contratarla.

- La integridad de la información debe ser de alta confiabilidad ya que está expuesto en el tráfico de internet.

- El desarrollo en cuanto a la seguridad debe ser alta y, por ende, debe generarse software embebido que ayude a enfrentar ataques informáticos a la que la información esté expuesta.

2.6.3. Estándar de los Servicios Web

Los servicios web pueden ser creados de varias maneras en múltiples plataformas y para lenguajes heterogéneos. Por este motivo el estándar de su uso y desarrollo queda a disposición del fabricante de software, no obstante, este tiene que tener en cuenta la disponibilidad, integridad y seguridad de la información a sus clientes e inclusive con la interacción a plataformas terceras (Pereira, 2003).

2.6.4. Tecnologías para el desarrollo de Servicios Web.

Como bien existen métodos para el desarrollo y acceso a los servicios web se ha destacado considerablemente la tecnología de extensión al lenguaje de hipertexto XML, el cual es definido por el usuario y para su uso debe ser soportado por un lenguaje de programación adicional ya sea como Java, Cobol, C++, etc.

"Una vez el documento XML es analizado, la información está disponible para ser utilizada por un cliente. Dado que XML es usado para la presentación, la comunicación, y la configuración, un cliente de un servicio web puede ser una aplicación, un web service o un humano" (Pereira, 2003).

2.6.5. Arquitectura del funcionamiento de un Servicio Web

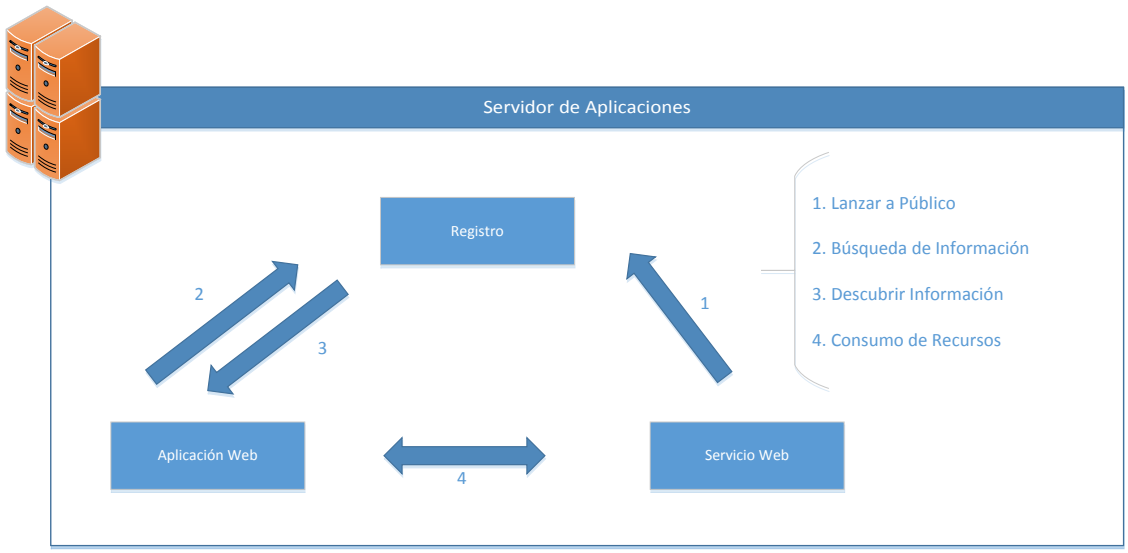


Ilustración II-8: Arquitectura para el funcionamiento de Servicios Web

Elaborado por: Diego Ponce – Juan Prado

Basado de*: (Pereira, 2003)*

En la Ilustración II-8 están implicadas las siguientes funcionalidades en cuanto respecta a servicios web:

1. El momento de generar o desarrollar un servicio web, éste debe ser registrado en el servidor de aplicaciones por medio del estándar UDDI (Descripción, Descubrimiento e Integración Universal) el cual es un registro público para levantar servicios web, así como también por medio del WSDL (Lenguaje de descripción de servicios web), el cual se presenta en archivos con formatos XML.

2. Ya sea por medio del usuario desarrollador, administrador o a petición de la misma plataforma de desarrollo, el servicio web es lanzado para su respectiva publicación en la cual quedará listada entre otros servicios web sea el caso, para su debida búsqueda y demás solicitudes de clientes.

3. Al momento de hacer una solicitud o llamada a un servicio web, el dispositivo cliente recibirá o bien la notificación de que ha sido procesada exitosamente o a su vez la información solicitada o el recurso deseado por la aplicación cliente.

4. El dispositivo cliente lanza por medio de internet o una red de telecomunicación, la petición al servidor de aplicación, para buscar y consumir de un predeterminado servicio web, del cual, debe recibir una respuesta.

2.6.6. Tecnologías de Servicios Web

Tomando en cuenta que los servicios web son componentes abiertos, adaptables a varias plataformas y tecnologías podemos clasificarlos en las siguientes:

- **SOAP (Protocolo de Acceso Simple a Objetos):** El método básico para el intercambio de recursos en un entorno de software distribuido.

- **WSDL (Lenguaje Descriptivo para servicios Web):** Es un protocolo que trabaja bajo documentos, en el cual se describe o enlista cada objeto o recurso que estará disponible dentro del servicio web al igual que los parámetros que requiere o dará como resultado al ser consumido. En este caso el desarrollador debe conocer como esta generado o implementado el servicio web, la estructura y sus objetos

descritos respectivamente para poder implementar el cliente, así mismo conocer la API que se está usando, y la disponibilidad en cuanto a los servicios del mismo.

"Si un web service se encuentra disponible al público, su WSDL también estará a la vista de cualquiera. Estos documentos XML pueden ser encontrados en registros UDDI o en web especializadas que funcionan como un buscador normal de internet, como por ejemplo www.xmethods.net *"* (Pereira, 2003).

- **UDDI (Descripción, Descubrimiento e Integración universal):** Como se ha indicado anteriormente, el servicio web para estar disponible a sus respectivos clientes debe ser registrado, y este se lo puede realizar por UDDI el cual es un BD en el cual estará publicado de manera universal los servicios web generados, y puede ser consumido mediante protocolos como por ejemplo SOAP[55].

- **WSDL (Lenguaje para inspección de servicios web):** Es un componente en el cual están contenidas una lista de todos los enlaces vinculados a los servicios web que ya han sido lanzados y publicados.

A diferencia de UDDI es que el primero mantiene la lista directa de o servicios web para un entorno abierto y distribuido, en cambio con WSDL, la lista de enlaces, es centralizada a ciertos entornos distribuidos.

[55] **SOAP:** Protocolo para el intercambio de información para el acceso simple de Objetos.

CAPÍTULO III.

DEFINICIÓN DEL SISTEMA

3.1. ANALISIS Y DISEÑO

En este capítulo recogeremos todos los requerimientos que nos permitirán desarrollar el aplicativo web y móvil para la monitorización de dispositivos a través de GPS y Google Cloud Messaging.

Para el desarrollo, se detallará a continuación todas las consideraciones tomadas para la elaboración exitosa del programa.

3.1.1. REQUERIMIENTOS

3.1.1.1. Identificación de actores

A partir de las reuniones con el líder de proyecto se ha podido identificar los siguientes usuarios los cuales intervienen directamente con las funcionalidades del sistema, estos son:

3.1.1.2. Administrador

El administrador es el encargado de manejar, configurar y crear todos los usuarios que utilizaran la aplicación móvil para el monitoreo y el chat. Los mismos podrán verificar las rutas en los que se mueven los usuarios y notificar a las personas pertinentes a través de un chat sobre su posicionamiento.

Además, el administrador será capaz de crear ubicaciones las cuales se las considera importantes para el usuario móvil, ya que son locaciones a las cuales el usuario debe llegar.

3.1.1.3. Usuario Android – Ios

El usuario Android – Ios es la persona previamente creada por el administrador. Este usuario tendrá la posibilidad de chatear tanto con los otros usuarios que tienen el mismo rol y con el administrador del sistema. Este va a ser monitoreado de acuerdo a los parámetros que el sistema le indique.

3.1.1.4. Historias de Usuario

"Las historias de usuario son descripciones simples y cortas que describen las funcionalidades que deberán ser implementadas en el software. Las historias son escritas por el propio cliente, con sus propias palabras y generalmente son registradas en tarjetas" (Fuentes, 2016).

La aplicación de monitorización que se ha desarrollado tiene dos partes importantes las cuales son: la primera es la aplicación de monitorización virtual administrador web que permitirá saber el posicionamiento y ubicación de los usuarios móviles y la comunicación

con los mismos a través de un chat, sin olvidarse de la creación y administración de usuarios; la segunda es el aplicativo de monitorización virtual móvil en dispositivos Android, que permitirá la monitorización valga la redundancia del posicionamiento de los usuarios que tengas la plataforma móvil y la comunicación entre usuarios sin importar el rol.

A continuación, se detallan las historias de usuario realizadas para la aplicación de monitorización.

3.1.1.5. Monitorización Virtual Administración

3.1.1.5.1. Autenticación de usuarios administrativos

Tabla III-1:Historia de Usuario 1: Autentificación de usuarios Administrativos

Historia de Usuario	
Número:01	**Nombre de Historia:** Autenticación de usuarios administrativos
Usuario: Administrador	
Prioridad en Negocio: Media ***(Alta/Media/Baja)***	**Riesgo en Desarrollo:** Media *(Alta/Media/Baja)*
Descripción: Los usuarios que accederán a la plataforma virtual en web serán identificados con el rol "Administrador". **Administrador:** El usuario administrador, para acceder a todos sus permisos y privilegios tendrá que autenticarse en la aplicación web con sus credenciales (nombre de usuario y contraseña). Las funcionalidades que tendrá este tipo de usuario es la gestión (crear, actualizar, eliminar y visualizar) catálogos necesarios para asignar niveles de acceso a los usuarios tanto a la plataforma web como también a los usuarios que accedan a la aplicación móvil (Administración de Usuarios), monitorización de usuarios, mensajería.	

Elaborado por: Diego Ponce – Juan Prado

3.1.1.5.2. Administración de Usuarios

Tabla III-2: Historia de Usuario 2: Administración de Usuarios

Historia de Usuario	
Número:02	**Nombre de Historia:** Administración de usuarios
Usuario: Administrador	
Prioridad en Negocio: Media ***(Alta/Media/Baja)***	**Riesgo en Desarrollo:** Media *(Alta/Media/Baja)*
Descripción: La administración de los usuarios es una de las partes más importantes dentro del sistema ya que en este proceso el administrador podrá crear, actualizar, eliminar y visualizar usuarios dependiendo el rol (Administrador, Android, Ios), y así el sistema pueda generar códigos de acceso los cuales serán enviados automáticamente vía email al usuario pertinente.	

Elaborado por: Diego Ponce – Juan Prado

3.1.1.5.3. Monitorización de usuarios

Tabla III-3: Historia de Usuario 3: Monitorización de Usuarios

Historia de Usuario	
Número:03	**Nombre de Historia:** Monitorización de usuarios.
Usuario: Administrador	
Prioridad en Negocio: Alta ***(Alta/Media/Baja)***	**Riesgo en Desarrollo:** Alta *(Alta/Media/Baja)*
Descripción: El sistema web o plataforma web debe permitir la monitorización de los usuarios móviles con rol Android, e Ios, para saber el posicionamiento histórico de los mismos en tiempo real. El administrador del sistema podrá crear marcadores o "PushPin's" dentro del mapa para que el usuario móvil sepa cuáles son los lugares importantes a los que debe ir.	

Elaborado por: Diego Ponce – Juan Prado

3.1.1.5.4. Mensajería Web

Tabla III-4: Historia de Usuario 4: Mensajería Web

Historia de Usuario	
Número:04	**Nombre de Historia:** Mensajería Web
Usuario: Administrador	
Prioridad en Negocio: Alta ***(Alta/Media/Baja)***	**Riesgo en Desarrollo:** Alta *(Alta/Media/Baja)*
Descripción: La plataforma web debe permitir rápidamente y eficazmente la comunicación con los usuarios móviles a través de un chat, sin riesgos de pérdida de información cuando el usuario esté en sesión o no.	

Elaborado por: Diego Ponce – Juan Prado

3.1.1.6. Monitorización Virtual Móvil

3.1.1.6.1. Autenticación de usuarios móviles

Tabla III-5: Historia de Usuario 5: Autenticación de usuarios Móviles

Historia de Usuario	
Número:05	**Nombre de Historia:** Autenticación de usuarios móviles
Usuario: Android	
Prioridad en Negocio: Media ***(Alta/Media/Baja)***	**Riesgo en Desarrollo:** Media *(Alta/Media/Baja)*
El usuario móvil, para acceder a las funcionalidades que tiene el aplicativo deberá iniciar sesión con sus credenciales (correo electrónico, contraseña). Las funcionalidades que tiene acceso este tipo de usuario es la de mensajería, como también las de ver los marcadores de posicionamiento a los cuales debe llegar, esto se visualizará a través de un mapa.	

Elaborado por: Diego Ponce – Juan Prado

3.1.1.6.2. Interface móvil monitorización

Tabla III-6: Historia de Usuario 6: Interface móvil de Monitorización

Historia de Usuario	
Número:06	**Nombre de Historia:** Interface móvil monitorización
Usuario: Android	**Riesgo en Desarrollo:** Baja
Prioridad en Negocio: Media ***(Alta/Media/Baja)***	*(Alta/Media/Baja)*
Descripción: La aplicación necesita tener una interfaz atractiva para el usuario, manejando una gama de colores y logos que tengan un significado simple y así el usuario se familiarice intuitivamente con la aplicación.	

Elaborado por: Diego Ponce – Juan Prado

3.1.1.6.3. Actualización contactos mensajería

Tabla III-7: Historia de Usuario 7: Actualización de contactos en mensajería

Historia de Usuario	
Número:07	**Nombre de Historia:** Actualización contactos mensajería
Usuario: Android	**Riesgo en Desarrollo:** Media
Prioridad en Negocio: Media ***(Alta/Media/Baja)***	*(Alta/Media/Baja)*
Descripción: La aplicación permitirá automáticamente al loguearse actualizar los contactos con los cuales se comunicará a través de un chat, pero una vez logueado el usuario también tendrá la funcionalidad de actualizar los mismos manualmente para saber si hay algún contacto nuevo sea administrador o un nuevo contacto con rol Android.	

Elaborado por: Diego Ponce – Juan Prado

3.1.1.6.4. Mensajería Móvil

Tabla III-8: Historia de Usuario 8: Mensajería Móvil

Historia de Usuario	
Número:08	**Nombre de Historia:** Mensajería Móvil
Usuario: Android	**Riesgo en Desarrollo:** Alta
Prioridad en Negocio: Media	*(Alta/Media/Baja)*

(Alta/Media/Baja)

Descripción:
El usuario tendrá la posibilidad de enviar mensajes a las diferentes plataformas sea Android o Web y a los usuarios pertinentes de cada una.

Elaborado por: Diego Ponce – Juan Prado

3.1.1.6.5. Visualización y ubicación mapa

Tabla III-9: Historia de Usuario 9: Visualización y ubicación de mapas

Historia de Usuario	
Número:09	**Nombre de Historia:** Visualización y ubicación mapa
Usuario: Android	**Riesgo en Desarrollo:** Alta
Prioridad en Negocio: Alta ***(Alta/Media/Baja)***	*(Alta/Media/Baja)*

Descripción:
El usuario tendrá la posibilidad de visualizar un mapa (Google Maps) con su ubicación actual y los lugares a los cuales deberá visitar, estos se mostrarán en el sistema según la parametrización que haya marcado el administrador previamente. El aplicativo deberá informar periódicamente al administrador el posicionamiento del mismo.

Elaborado por: Diego Ponce – Juan Prado

3.1.2. ANÁLISIS

Ya definido todas las funcionalidades de la aplicación de monitoreo tanto en la parte web como móvil con el levantamiento de las historias de usuario, es necesario un análisis que nos permita evaluar los tiempos que el desarrollo requiere para terminar adecuadamente el proyecto de la forma más eficiente y exitosa posible. Esta estimación y análisis ayudará a cumplir a cabalidad las fechas propuestas con los recursos necesarios.

3.1.2.1. Estimación de historias de usuario

Con las historias de usuario ya definidas con el líder del proyecto es necesario que los programadores tomando en cuenta todos los recursos disponibles estimen los tiempos necesarios en base a la información propuesta.

Ahora bien, tomando en cuenta el esfuerzo para el desarrollo de las aplicaciones web y móvil de monitorización es necesario tomar en cuenta los siguientes parámetros:

- Numero de desarrolladores: 2
- Horas diarias de desarrollo (sin distracciones): 5
- Días semanales de desarrollo (sin distracciones): 6

Ya con estos parámetros la fórmula que utilizamos da el tiempo de desarrollo ideal por semana:

$$2\ personas \times 5\ horas \times 6\ días\ = 60\ horas$$

La tabla III-10 refleja el tiempo ideal para el desarrollo de las aplicaciones el cual es 60 horas en una semana de trabajo.

Tabla III-10: Estimación de Historias de Usuario

Estimación de Historias de Usuario				
No.	**Título**	**Tiempo ideal**	**Riesgo en desarrollo**	**Horas**
1	Autenticación de usuarios administrativos	0.17	Alta	10 horas
2	Administración de usuarios	0.42	Alta	25 horas
3	Monitorización de usuarios.	0.67	Media	40 horas
4	Mensajería Web	1.33	Media	80 horas
5	Autenticación de usuarios móviles	0.17	Alta	10 horas
6	Interface móvil monitorización	0.5	Baja	30 horas

7	Actualización contactos mensajería	0.67	Media	40 horas
8	Mensajería Móvil	1.5	Media	90 horas
9	Visualización y ubicación mapa	1.67	Media	100 horas

Elaborado por: Diego Ponce – Juan Prado

3.1.2.2. Priorización de Historias de Usuario

Una vez ya definido la estimación de tiempo que lleva a cabo cada una de las historias de usuario continuamos priorizando cada una de ellas.

La priorización nos permitirá identificar cuáles de las historias de usuario son más importantes, la escala que nos ayudara a entender esto son:

- Alta: Las historias que contengan este nivel tendrán que ser implementadas inmediatamente.

- Media: Las historias con este nivel podrán esperar, pero tendrán que ser implementadas una vez finalizadas las historias con nivel alto.

- Baja: Son las historias que se tomarán en cuenta al último y podrán esperar un largo plazo.

La Priorización está reflejada en la tabla III-11 mostrada a continuación.

Tabla III-11: Priorización de Historias de Usuario

Priorización de Historias de Usuario					
Alta		Media		Baja	
No.	**Título**	**No.**	**Título**	**No.**	**Título**
1	Autenticación de usuarios administrativos	3	Monitorización de usuarios.	**6**	Interface móvil monitorización
2	Administración de usuarios	4	Mensajería Web		
5	Autenticación de usuarios móviles	7	Actualización contactos mensajería		
		8	Mensajería Móvil		
		9	Visualización y ubicación mapa		

Elaborado por: Diego Ponce – Juan Prado

3.1.2.3. Plan de entregas

Luego de haber identificado las prioridades que tienen cada historia de usuario es necesario definir los planes de entrega con su respectiva iteración las mismas que se agruparán de tal forma que especifiquen las fechas en las cuales se entregarán cada una de ellas a los usuarios como se muestra en la Tabla III-12.

Para el proyecto Monitorización Virtual se han definido las siguientes iteraciones las cuales se detallan a continuación:

1. En la primera iteración se abarcará todas las historias de usuario que tengan que ver con las administraciones de los usuarios tanto en la parte web como móvil, para poder avanzar con las siguientes historias de usuario que dependen directamente de ellas.

2. La segunda iteración contemplará todas las historias de usuario que tengan que ver con la configuración de GCM (Google Cloud Messaging) y las notificaciones en tiempo real entre el servidor y el aplicativo móvil.

3. La última iteración la cual es la tercera comprende todo lo que es la monitorización del dispositivo móvil y como las mismas enviarán notificaciones a través de GCM, y la visualización de la aplicación para el usuario.

Tabla III-12: Plan de Entregas e iteraciones

Plan de Entregas e iteraciones					
No.	**Historia de Usuario**	**Tiempo Ideal**	**Iteración asignada**	**Fecha en la que se entregará**	**Entrega Asignada**
1	Autenticación de usuarios administrativos	0.17	1	08/08/2016	1
2	Administración de usuarios	0.42	1	22/08/2016	1
5	Autenticación de usuarios móviles	0.17	1	10/09/2016	1

7	Actualización contactos mensajería	0.17	2	17/09/2016	2
4	Mensajería Web	1.33	2	29/09/2016	2
8	Mensajería Móvil	1.5	2	08/10/2016	2
3	Monitorización de usuarios.	0.67	3	22/10/2016	2
9	Visualización y ubicación mapa	1.67	3	05/11/2016	3
6	Interface móvil monitorización	0.5	3	19/11/2016	3

Elaborado por: Diego Ponce – Juan Prado

3.1.3. ARQUITECTURA DE LA SOLUCIÓN

En la Ilustración III-1 se puede observar la arquitectura de comunicación, estructura y funcionamiento entre los servidores de base de datos, servidores de Google Cloud Messaging (GCM), el aplicativo web y móvil.

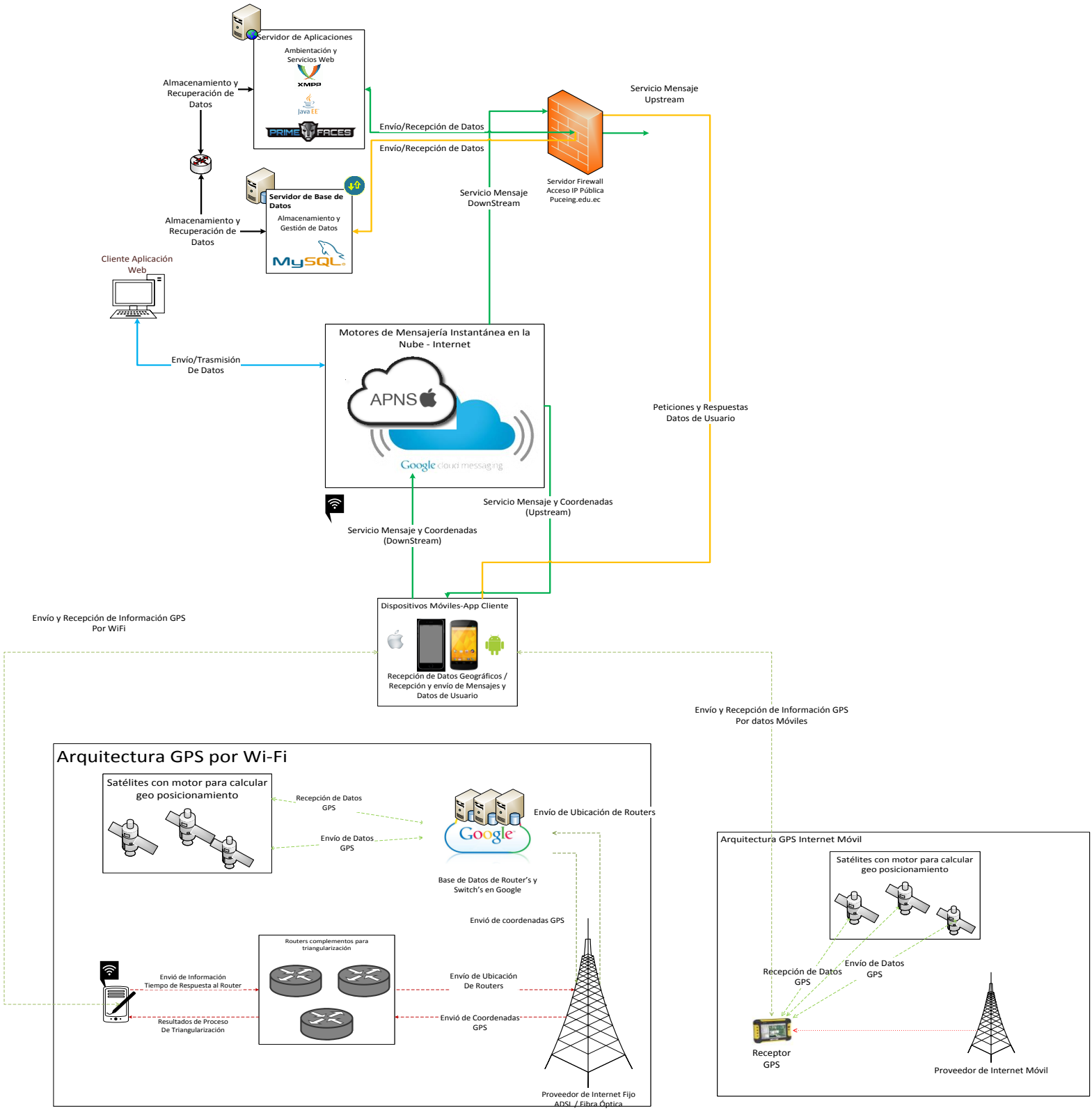


Ilustración III-1: Arquitectura a la Solución del Proyecto de Software

Elaborado por: Diego Ponce – Juan Prado

3.1.3.1. Requerimientos de hardware y Software para Monitorización Virtual Móvil

El aplicativo para el monitoreo y la ubicación de dispositivos móviles requiere ciertas medidas de tecnología mínima en lo que respecta su entorno y adecuado funcionamiento, lo cual implica equipo de hardware y software con mínimas prestaciones y versiones de los mismos respectivamente.

Tabla III-13: Requerimientos para Aplicación Móvil

Requerimientos para Aplicación Móvil		
Categoría	**Descripción**	**Características**
Hardware	Equipo de telefonía Móvil Smartphone	Chip Wi-Fi 802.11 a/b/g/n/ac
		Acceso a Internet Móvil
		Mem. Ram: 1GB en adelante
		Procesador: ARM Dual Core a 1 Ghz min.
		Chip Antena A-GPS, GLONASS
		Batería: 1500 mAh. ó más
		Pantalla 4' pulgadas min.
		Almacenamiento: 20 mb. Min.
Software	Sistema Operativo Android	Versión 4.0.x. Ice Cream Min.
	Servicios de Google Play	Versión 9.0.1. Min

Elaborado por: Diego Ponce – Juan Prado

3.1.3.2. Requerimientos de hardware y Software para Monitorización Virtual Administrador

Al momento de implementar una infraestructura de servidores ya sea para aplicaciones, base de datos, cortafuegos, etc., para una empresa o entidad corporativa se pone en evaluación medidas de consumo de usuarios como pueden ser banda consumida por sesión, número de solicitudes en paralelo, consumo de memoria y porcentaje de procesamiento, etc., todos estos datos son evaluados para proyectar la adquisición de hardware y completar el proceso de instalación de una infraestructura de servidores.

No obstante, el alcance de este trabajo es un prototipo listo para la implantación de software, para cualquier entidad y lógica de negocio o funcionamiento, por tal motivo se define medidas de consumo mínimas predeterminadas con su respectivo hardware y software mínimo recomendable para el adecuado funcionamiento de la aplicación.

Tabla III-14:Parámetros de medidas de consumo mínimos para los servidores

No.	Descripción de Recurso	Valor/Unidad
1	Usuarios registrados	50
2	Administradores Registrados	10
3	Solicitudes diarias por usuario	100
4	Sesiones simultáneas	50
5	Consumo Diario Internet por aplicación web	500 mb/Dia
6	Escritura de Disco Duro Diario	50mb/Día
7	Promedio RAM diaria consumida	2GB/Día

Elaborado por: Diego Ponce – Juan Prado

El Hardware y Software recomendado para los requerimientos anteriormente mencionados está representado en la siguiente tabla.

Tabla III-15: Requerimientos para Plataforma Web

Requerimientos para Plataforma Web		
Categoría	**Descripción**	**Características**
Hardware	Servidor de Aplicaciones	Procesador Intel Xeon Dual Core a 1.6 Ghz min.
		Tarjeta Fast Ethernet: 1 min.
		Memoria Cache: 1mb min.
		Memoria RAM: 6 GB min.
		Disco Duro: 30GB. Min. Con Tecnologia RAID
Software	Sistema Operativo Windows (Opcional)	Windows Server Versión 2012 min.
	Sistema Operativo Linux (Opcional)	Linux Centos Versión 6 min.
	Máquina virtual Java	JVM Versión 7.x en adelante
	Entorno de Ejecución de Java	JRE versión 7.x en adelante
	Software para escritorio remoto	Team Wiever Versión 11 en adelante
	Navegador Web	Navegador Mozilla Firefox, Google Chrome Actualizados
	Servidor de Aplicaciones Jboss	Jboss AS 7.x en adalente
Hardware	Servidor de Base de datos	Procesador Intel Xeon Dual Core a 1.6 Ghz min.
		Tarjeta Fast Ethernet: 1 min.
		Memoria Cache: 1mb min.
		Memoria RAM: 2 GB min.
		Disco Duro: 1TB. Min. Con Tecnologia RAID

Software	Sistema Operativo Windows (Opcional)	Windows Server Versión 2012 min.
	Sistema Operativo Linux (Opcional)	Linux Centos Versión 6 min.
	Máquina virtual Java	JVM Versión 7.x en adelante
	Entorno de Ejecución de Java	JRE versión 7.x en adelante
	Software para escritorio remoto	Team Wiever versión 11 en adelante
	Motor de Base de Datos Mysql	Versión 4.x en adelante
Hardware	Terminal para Aplicación Administrador	Procesador Intel o AMD Dual Core a 1 Ghz min.
		Memoria Cache: 1mb
		Memoria RAM: 1GB. Min
		Disco duro: 30 GB min.
		NIC de comunicación: 1 min. con acceso a internet
Software	Sistema Operativo Windows (Opcional)	Windows 7 SP1 en adelante
	Sistema Operativo Linux (Opcional)	Sistemas Linux con escritorio GNOME, KDE o compatible
	Máquina virtual Java	JVM Versión 7.x en adelante
	Entorno de Ejecución de Java	JRE versión 7.x en adelante
	Navegador Web	Navegador Mozilla Firefox, Google Chrome Actualizados

Elaborado por: Diego Ponce – Juan Prado

3.1.4. DICCIONARIO DE DATOS

Como parte del sistema es necesario conocer como está estructurado el diccionario de datos que compone la base de datos la misma que es una de las partes más importantes de la funcionalidad del proyecto.

Se presenta a continuación una de las tablas de la base de datos para registro de posicionamiento del usuario el cual fue enviado previamente por el dispositivo. Se explica además el funcionamiento que realiza cada campo en la base de datos.

3.1.4.1. Tabla Coordinates

Tabla III-16: Diccionario de Datos para la Tabla "Coordinates"

Nombre	**Coordinates**			
Descripción	Tabla donde se almacena el posicionamiento enviado por el dispositivo Android del usuario			
	Atributos			
coo_id	**Tipo**	**PK**	**NN**	**AI**
	Int	☒	☒	☒
	Identificador único de cada fila insertada en la tabla.			
usr_id	**Tipo**	**FK**	**NN**	**AI**
	Int	☒	☒	☐
	ID del usuario del dispositivo móvil.			
coo_latitude	**Tipo**	**FK**	**NN**	**AI**
	Decimal(50,30)	☐	☒	☐
	Latitud de posicionamiento enviado por el dispositivo móvil			
coo_longitude	**Tipo**	**PK**	**NN**	**AI**
	Decimal(50,30)	☐	☒	☐
	Longitud de posicionamiento enviado por el dispositivo móvil			
coo_date_creation	**Tipo**	**PK**	**NN**	**AI**
	Datetime	☐	☒	☐
	Fecha de recepción de las coordenadas del dispositivo móvil			

Elaborado por: Diego Ponce – Juan Prado

Por la numerosa cantidad de tablas que componen el modelo de la base de datos de este sistema mostraremos únicamente la tabla anteriormente presentada para entender cómo se manejan los diccionarios de datos.

3.1.5. CASOS DE USO

"El modelado de casos de uso es un método orientado a los usuarios para identificar necesidades funcionales de un nuevo sistema de información. El modelado de casos de uso es una técnica que permite modelar las funciones de un sistema en términos de eventos, de quien inicia los eventos y de cómo responde el sistema a estos eventos." (Alarcón, 2006)

Como parte del proyecto se procederá a continuación con los diagramas de caso de uso del sistema Monitorización, tomando en cuenta los actores que se identificaron y la descripción de cada evento en los que los mismos participan.

3.1.5.1. Diagrama casos de uso: Usuarios

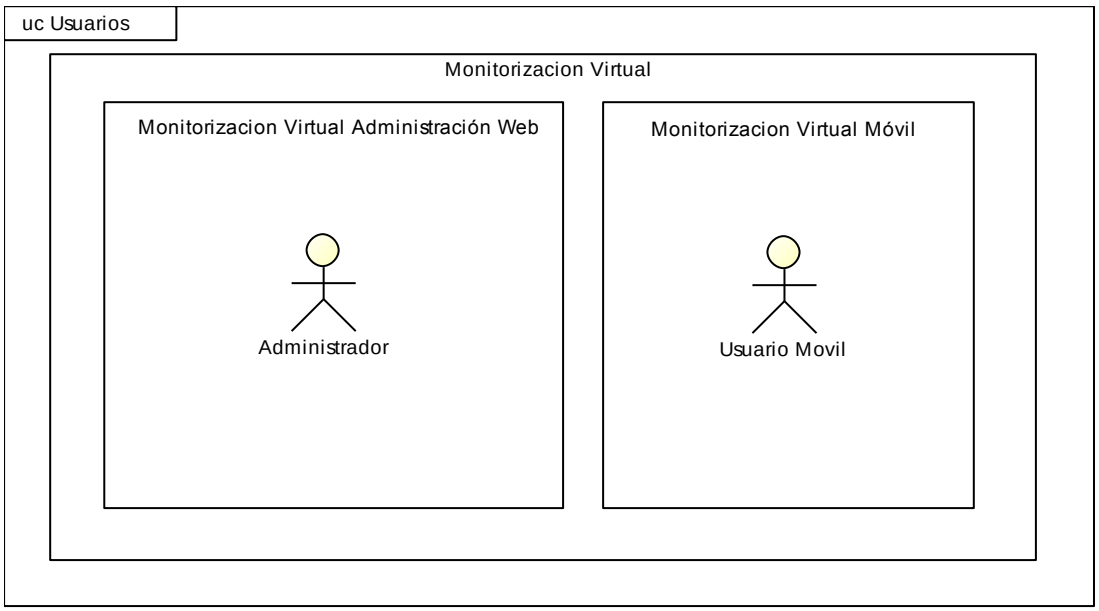


Diagrama III-1: Diagrama de Caso de Uso: Usuarios

Elaborado por: Diego Ponce – Juan Prado

Para el desarrollo de este sistema como podemos observar se contemplan dos entornos o ambientes: el primero es el de Monitorización Virtual Administración web; el segundo Monitorización Virtual Móvil, que será específicamente para usuarios con dispositivos con plataforma Android. Tal como se muestra en el Diagrama III-1.

3.1.5.2. Diagrama de casos de uso: Monitorización Virtual Móvil

3.1.5.2.1. Usuario Móvil

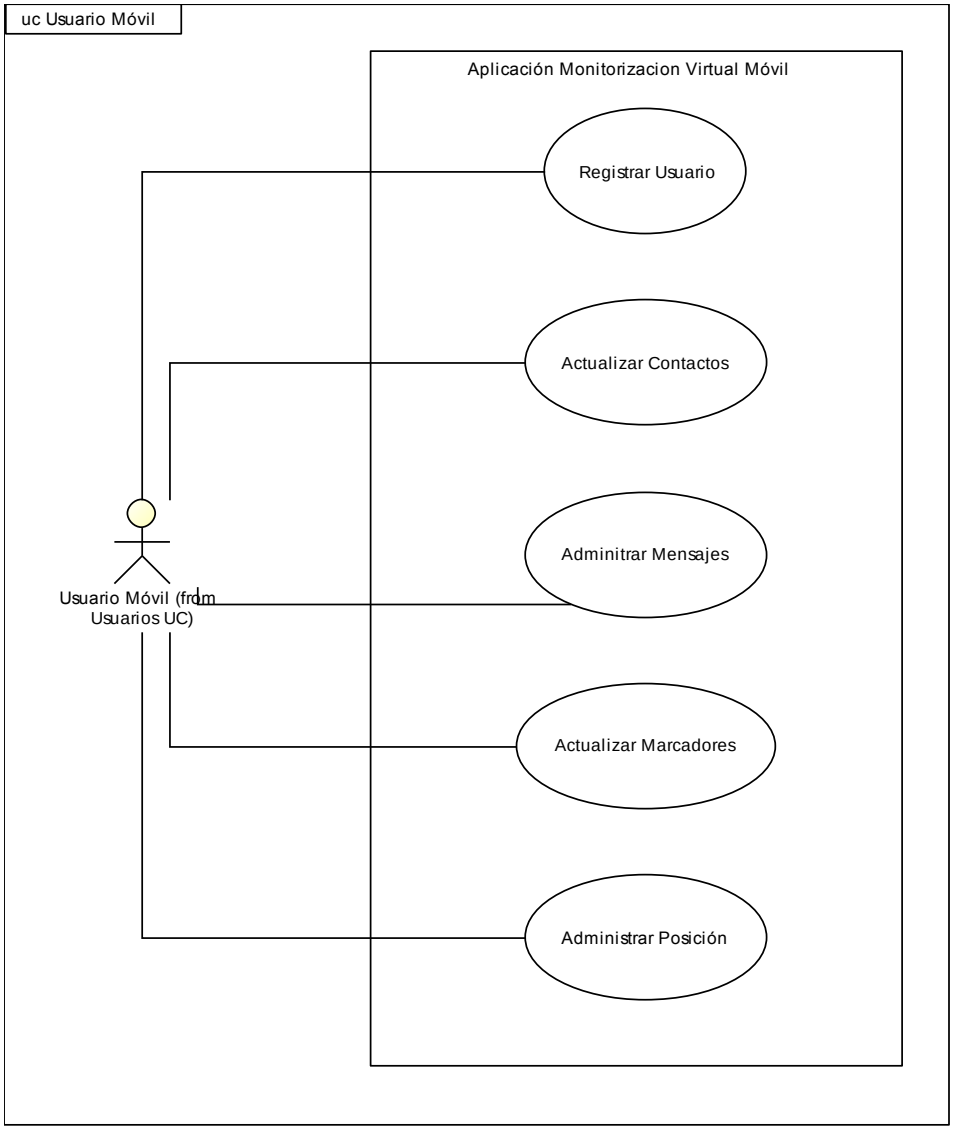


Diagrama III-2: Diagrama de caso de Uso: Usuario Móvil

Elaborado por: Diego Ponce – Juan Prado

3.1.5.3. Diagrama de casos de uso: Monitorización Virtual Administración Web

3.1.5.3.1. Administrador

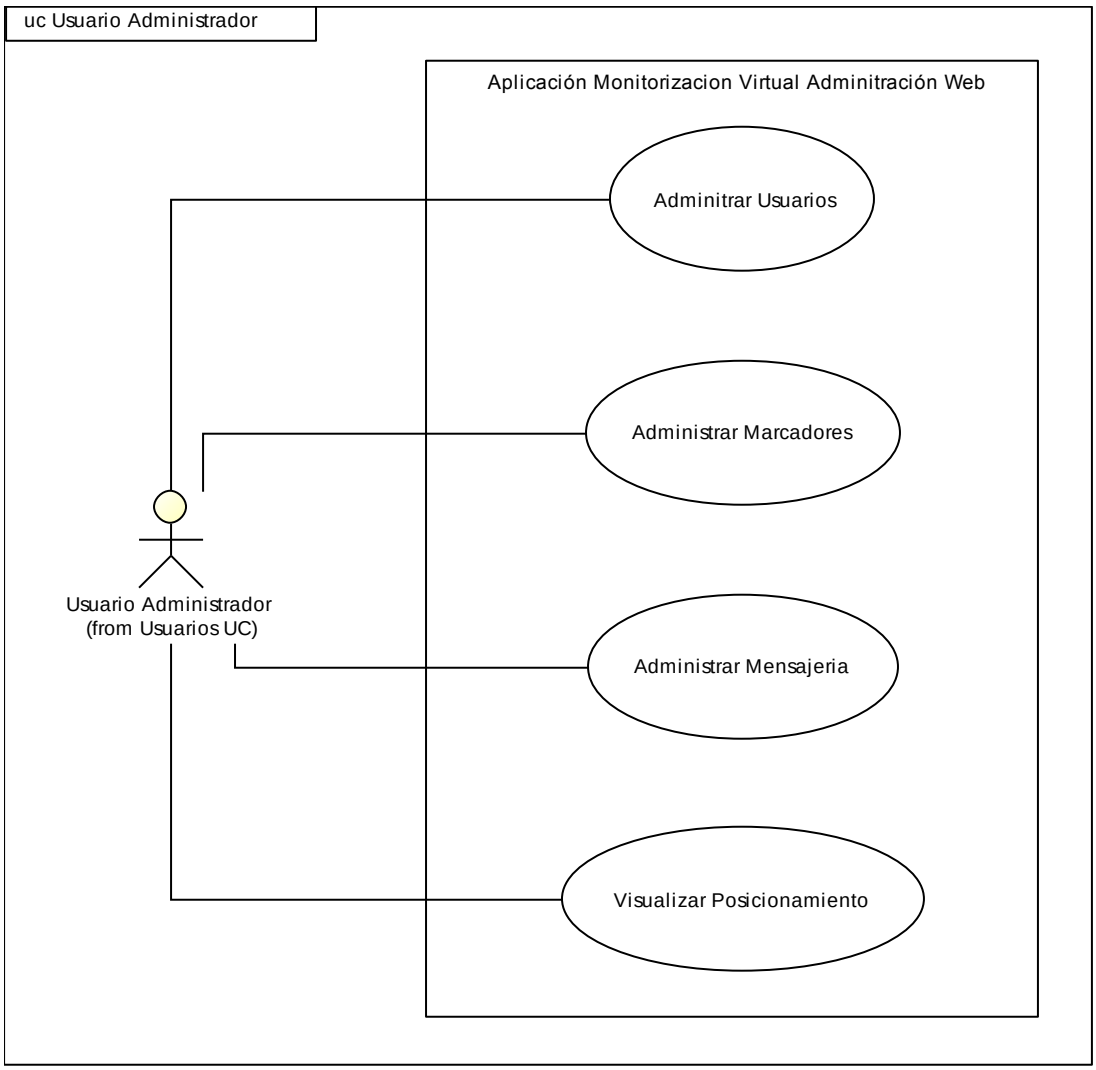


Diagrama III-3: Diagrama de Caso de Uso: Administrador

Elaborado por: Diego Ponce – Juan Prado

3.1.5.4. Descripción casos de uso: Monitorización Virtual Móvil.

3.1.5.4.1. Registrar Usuario

Tabla III-17: Descripción de Caso de Uso 1: Registrar Usuario

Nombre:	**Registrar Usuario**			
Identificador:	CU-01			
Actor:	Usuario Móvil			
Descripción:	Consiste en un formulario de registro que permite crear una cuenta en la aplicación monitorización virtual QR móvil.			
Precondiciones:	1. Tener una cuenta de correo activa. 2. Tener código de registro enviado previamente al correo electrónico del usuario			
Post-condiciones:	1. Se guarda el token de GCM (Google Cloud Messaging) de la nueva cuenta en la base de datos.			
Flujo principal:	**ACTOR**		**SISTEMA**	
	1	El usuario ingresa a la pantalla principal de la aplicación.	**2**	El sistema mostrará un formulario que solicita los campos: correo electrónico y el código de registro de la cuenta.
	3	El usuario registra los datos presentes en el formulario	**5**	El sistema verifica que los datos ingresados estén completos y sean correctos.
	4	El usuario da clic en el botón registrar.	**6**	El sistema guarda el token generado por GCM en la base de datos, y se envía a web service.
			7	El sistema muestra un mensaje de éxito e ingresa a la pantalla principal de la aplicación.
Flujo alternativo			**3.1**	Si existe algún error de validación de los campos del formulario sistema mostrará un mensaje de que los datos están mal ingresados en el formulario.
			4.1	Si los datos no coinciden con la consulta a la base de datos, el sistema mostrará mensaje de datos incorrectos.
			7.1	Si existen errores en el punto 3.1 o 4.1 ir al paso 3 del flujo principal.
Resultado:	Se creará la cuenta en la aplicación monitorización virtual móvil, para poder empezar el monitoreo del dispositivo y del usuario y mensajería instantánea.			

Elaborado por: Diego Ponce – Juan Prado

3.1.5.4.2. Actualizar contactos

Tabla III-18: Descripción de Caso de Uso 2: Actualizar Contactos

Nombre:	**Actualizar Contactos**			
Identificador:	CU-02			
Actor:	Usuario móvil			
Descripción:	Consiste en un formulario donde aparecerán todos los usuarios con perfil Administrador y Android para poder acceder al chat.			
Precondiciones:	1. Tener cuenta activa. 2. Tener token de GCM. 3. Tener Internet activo.			
Post-condiciones:				
Flujo principal:	**ACTOR**		**SISTEMA**	
	1	El usuario ingresa a la pantalla principal de la aplicación.	**2**	El sistema mostrará un formulario sin contactos.
	3	El usuario da clic en actualizar contactos.	**4**	El sistema consumirá el servicio web correspondiente a los usuarios que se encuentran en la base de datos.
			5	El sistema guarda los contactos en la base local del dispositivo.
			6	El sistema muestra un mensaje de éxito y actualiza la vista con los contactos.
Flujo alternativo			**4.1**	Si el internet no está activado en el celular, se mostrará un mensaje para que se encienda el consumo de Wi-Fi o plan de datos. Ir al paso 3
			4.2	Si existe un problema con el consumo del servicio web, se mostrará mensaje de error en problemas del servidor del aplicativo web. Ir al paso 3.
			5.1.	Si ya existen contactos, el sistema procederá a actualizar el token GCM de los contactos, o si existe algún usuario nuevo procederá a ingresar en la base de datos local del dispositivo. Ir al paso 6.
Resultado:	Se mostrarán todos los contactos disponibles en el formulario para poder empezar la mensajería instantánea con los mismos.			

Elaborado por: Diego Ponce – Juan Prado

3.1.5.5. Administrar mensajes

3.1.5.5.1. Enviar mensaje

Tabla III-19: Descripción de Caso de Uso 3: Enviar Mensaje

Nombre:	**Enviar Mensaje**			
Identificador:	CU-03			
Actor:	Usuario móvil			
Descripción:	Consiste en un formulario donde podrá el usuario enviar mensajes a los contactos presentes en su lista.			
Precondiciones:	1. Haberse registrado en el aplicativo previamente. 2. Tener contactos en la base local del dispositivo. 3. Tener token actualizado de GCM.			
Post-condiciones:				
Flujo principal:	**ACTOR**		**SISTEMA**	
	1	El usuario ingresa a la pantalla principal de la aplicación.	**3**	El sistema mostrará un formulario con contactos.
	2	El usuario ingresa al menú principal y da clic sobre el dibujo de mensajería.	**5**	El sistema mostrará el formulario de mensajería.
	4	El usuario escogerá el contacto al cual quiere enviar un mensaje y dará clic sobre el mismo.	**7**	El sistema guardará el mensaje localmente y enviara el mensaje através de GCM.
	6	El usuario ingresa el mensaje y dará clic en ENVIAR.	**8**	El sistema mostrará el mensaje en pantalla.
Flujo alternativo			**5.1**	Si ya existen mensajes entre el contacto y el usuario el aplicativo consultará la base de datos local y mostrará en pantalla los mensajes entre los mismos.
			7.1	Si el servidor no devuelve un mensaje ACT (el usuario recibió el mensaje) en el dispositivo, el sistema guardará el mensaje como pendiente y volverá a enviar el mensaje en un lapso de tiempo.
			7.2.	Si el dispositivo no tiene internet, el mensaje se guardará como pendiente y volverá a enviar el mismo en un lapso de tiempo.
Resultado:	Se enviarán los mensajes al usuario y mostrará en pantalla con espera a una respuesta.			

Elaborado por: Diego Ponce – Juan Prado

3.1.5.5.2. Recibir Mensaje

Tabla III-20: Descripción de Caso de Uso 4: Recibir Mensaje

Nombre:	**Recibir Mensaje**		
Identificador:	CU-04		
Actor:	Usuario móvil		
Descripción:	Consiste en un formulario donde podrá recibir mensajes de los contactos presentes en su lista.		
Precondiciones:	1. Haberse registrado en el aplicativo previamente. 2. Tener contactos en la base local del dispositivo. 3. Tener token actualizado de GCM. 4. Tener internet activo.		
Post-condiciones:			
Flujo principal:	**ACTOR**		**SISTEMA**
	5 El usuario dará clic en el usuario correspondiente.	**1**	El sistema recibirá el mensaje.
		2	El sistema identificará de quien es el mensaje recibido y guardará el mensaje localmente.
		3	El sistema devolverá un mensaje ACT (recibido correctamente) al usuario correspondiente.
		6	El sistema mostrará el mensaje.
Flujo alternativo		**5.1**	Si la aplicación está en background el sistema mostrará una notificación con el mensaje y el usuario correspondiente.
		6.1	Si la aplicación no está en background el sistema mostrará el mensaje en el formulario actualizando los mensajes.
Resultado:	Se recibirán los mensajes del contacto y mostrará en pantalla con espera a una respuesta.		

Elaborado por: Diego Ponce – Juan Prado

3.1.5.6. Actualizar Marcadores.

Nombre:	**Actualizar Marcadores**			
Identificador:	CU-05			
Actor:	Usuario móvil			
Descripción:	Consiste en un formulario donde aparecerá el mapa de Google Maps con un botón para actualizar marcadores de referencia para el usuario.			
Precondiciones:	1. Tener cuenta activa. 2. Tener internet activo.			
Post-condiciones:				
Flujo principal:	**ACTOR**		**SISTEMA**	
	1	El usuario ingresa a la pantalla principal de la aplicación.	**2**	El sistema mostrará el formulario principal.
	3	El usuario da clic en botón con imagen de mapa.	**4**	El sistema mostrará el formulario correspondiente al mapa.
	5	El usuario dará clic sobre el botón con imagen de push pin.	**6**	El sistema guarda las posiciones de los marcadores en base local.
			7	El sistema muestra un mensaje de éxito y actualiza la vista con los marcadores.
Flujo alternativo			**4.1**	Si el internet no está activado en el celular, se mostrará un mensaje para que se encienda el consumo de Wi-Fi o plan de datos. Ir al paso 3
			6.1	Si existe un problema con el consumo del servicio web, se mostrará mensaje de error en problemas del servidor del aplicativo web. Ir al paso 5.
			5.1.	Si ya existen marcadores, el sistema procederá a actualizarlos. Ir al paso 6.
Resultado:	Se mostrarán todos marcadores de referencia para el usuario en un mapa de Google Maps.			

Elaborado por: Diego Ponce – Juan Prado

3.1.5.7. Administrar posición

Tabla III-21: Descripción de Caso de Uso 6: Administrar Posición

Nombre:	**Administrar posición**			
Identificador:	CU-06			
Actor:	Usuario móvil			
Descripción:	Consiste en que el sistema envié periódicamente el posicionamiento del usuario.			
Precondiciones:	1. Tener cuenta activa. 2. Tener internet activo. 3. Tener token GCM actualizado			
Post-condiciones:				
Flujo principal:	**ACTOR**		**SISTEMA**	
	1	El usuario ingresa a la pantalla principal de la aplicación.	**2**	El sistema mostrará el formulario principal.
	3	El usuario da clic en botón con imagen de mapa.	**4**	El sistema mostrará el formulario correspondiente al mapa.
			5	El sistema consumirá internet y mostrara el posicionamiento actual del usuario
			6	El sistema enviará por GCM un mensaje al administrador informándole de su posición
Flujo alternativo			**6.1**	Si el servidor no devuelve un mensaje ACT (el usuario recibió el mensaje), el sistema guardara el mensaje como pendiente y volverá a enviar el mensaje en un lapso de tiempo.
			6.2.	Si el dispositivo no tiene internet, el mensaje se guardará como pendiente y volverá a enviar el mismo en un lapso de tiempo.
Resultado:	Se mostrará la posición actual del usuario además de notificar al administrador su ubicación.			

Elaborado por: Diego Ponce – Juan Prado

3.1.5.8. Descripción casos de uso: Monitorización Virtual Administración Web.

3.1.5.8.1. Administrar usuarios

3.1.5.8.1.1. Ingresar Usuario

Tabla III-22: Descripción de Usuario 7: Ingresar Usuario

Nombre:	**Administrar usuarios – Ingresar Usuario**			
Identificador:	CU-07			
Actor:	Usuario administrador.			
Descripción:	Consiste en un formulario que permite el registro de los usuarios con su respectivo rol para poder ingresar a la aplicación web o móvil.			
Precondiciones:	1. Tener cuenta con perfil administrador.			
Post-condiciones:				
Flujo principal:	**ACTOR**		**SISTEMA**	
	1	El usuario ingresa a la pantalla principal de la aplicación.	**2**	El sistema mostrará el formulario principal.
	3	El usuario da clic en opción usuarios del menó principal.	**4**	El sistema mostrará en una tabla los usuarios registrados previamente en el aplicativo web
	5	El usuario da clic en nuevo usuario.	**6**	El sistema muestra dialogo con campos necesarios para crear un usuario.
	7	El usuario ingresa los campos: nickname, name, lastname, email, password, rol del usuario.		
	8	El usuario da clic en guardar.	**9**	El sistema guarda el usuario en la base de datos y actualiza la tabla de usuarios en la vista de la pantalla principal de usuarios
			10	El sistema enviará email al usuario correspondiente mostrándole sus datos para ingreso a los aplicativos correspondientes.
Flujo alternativo			**8.1**	Si los campos requeridos por la aplicación no están llenos o no cumplen las validaciones el sistema mostrará mensaje de error especificando lo anteriormente.
			9.1	Si el rol es Android el sistema generara un código

				de registro que se adjuntara el email que será enviado luego de haber guardado el usuario.
Resultado:	Se registrará el usuario correspondiente en la base de datos y se notificará mediante email los datos para ingreso del usuario.			

Elaborado por: Diego Ponce – Juan Prado

3.1.5.8.1.2. Modificar Usuario

Tabla III-23: Descripción de Caso de Uso 8: Modificar Usuario

Nombre:	**Administrar usuarios – Modificar Usuario**			
Identificador:	CU-08			
Actor:	Usuario administrador			
Descripción:	Consiste en un formulario que permite modificar los usuarios que fueron registrados previamente.			
Precondiciones:	1. Tener cuenta con perfil administrador.			
Post-condiciones:				
Flujo principal:	**ACTOR**		**SISTEMA**	
	1	El usuario ingresa a la pantalla principal de la aplicación.	**2**	El sistema mostrará el formulario principal.
	3	El usuario da clic en opción usuarios del menú principal.	**4**	El sistema mostrará en una tabla los usuarios registrados previamente en el aplicativo web.
	5	El usuario da clic en editar el usuario.	**6**	El sistema muestra diálogo con datos del usuario.
	7	El usuario actualiza los campos: nickname, name, lastname, email, password, rol del usuario.		
	8	El usuario da clic en guardar.	**9**	El sistema guarda el usuario en la base de datos y actualiza la tabla de usuarios en la vista de la pantalla principal de usuarios
Flujo alternativo			**8.1**	Si los campos requeridos por la aplicación no están llenos o no cumplen las validaciones el sistema mostrará mensaje de error especificando lo anteriormente.
Resultado:	Se actualizará el usuario correspondiente en la base de datos.			

Elaborado por: Diego Ponce – Juan Prado

3.1.5.8.1.3. Eliminar Usuario

Tabla III-24: Descripción de Caso de Uso 9: Eliminar Usuario

Nombre:	**Administrar usuarios – Eliminar Usuario**			
Identificador:	CU-09			
Actor:	Usuario administrador			
Descripción:	Consiste en un formulario que permite eliminar los usuarios que fueron registrados previamente.			
Precondiciones:	1. Tener cuenta con perfil administrador.			
Post-condiciones:				
Flujo principal:	**ACTOR**		**SISTEMA**	
	1	El usuario ingresa a la pantalla principal de la aplicación.	**2**	El sistema mostrará el formulario principal.
	3	El usuario da clic en opción usuarios del menú principal.	**4**	El sistema mostrará en una tabla los usuarios registrados previamente en el aplicativo web.
	5	El usuario da clic en eliminar usuario.	**6**	El sistema elimina el usuario en la base de datos y actualiza la tabla de usuarios en la vista de la pantalla principal de usuarios
Flujo alternativo				
Resultado:	Se eliminará el usuario correspondiente en la base de datos.			

Elaborado por: Diego Ponce – Juan Prado

3.1.5.8.1.4. Consultar Usuario

3.1.5.8.1.4.1. Consulta General

Tabla III-25: Descripción de Caso de Uso: Consulta General

Nombre:	**Administrar usuarios – Consulta General**			
Identificador:	CU-10			
Actor:	Usuario administrador			
Descripción:	Consiste en un formulario que permite consultar los usuarios que fueron registrados previamente.			
Precondiciones:	1. Tener cuenta con perfil administrador.			
Post-condiciones:				
Flujo principal:	**ACTOR**		**SISTEMA**	
	1	El usuario ingresa a la pantalla principal de la aplicación.	**2**	El sistema mostrará el formulario principal.
	3	El usuario da clic en opción usuarios del menú principal.	**4**	El sistema mostrará el formulario de usuarios
	5	El usuario da clic en buscar.	**6**	El sistema actualizará tabla de contactos mostrando todos los contactos valga la redundancia sin ningún filtro.
Flujo alternativo				
Resultado:	Se mostrará todos los contactos en una tabla.			

Elaborado por: Diego Ponce – Juan Prado

3.1.5.8.1.4.2. Consulta por parámetro

Tabla III-26: Descripción de Caso de Usuario 11: Consulta por Parámetro

Nombre:	**Administrar usuarios – Consulta por parámetro**			
Identificador:	CU-11			
Actor:	Usuario administrador			
Descripción:	Consiste en un formulario que permite consultar por parámetros los usuarios que fueron registrados previamente.			
Precondiciones:	1. Tener cuenta con perfil administrador.			
Post-condiciones:				
Flujo principal:	**ACTOR**		**SISTEMA**	
	1	El usuario ingresa a la pantalla principal de la aplicación.	**2**	El sistema mostrará el formulario principal.
	3	El usuario da clic en opción usuarios del menú principal.	**4**	El sistema mostrará el formulario de usuarios
	5	El usuario ingresara cédula o apellido del usuario a buscar y da clic en buscar.	**6**	El sistema consulta a la base de datos.
			7	El sistema actualizará tabla de contactos mostrando los

			contactos que cumplan esa condición de búsqueda.
Flujo alternativo		**6.1**	Sistema muestra mensaje de error: no existe usuario buscado.
Resultado:	Se mostrará contactos que cumplan una condición ingresados en los parámetros de búsqueda.		

Elaborado por: Diego Ponce – Juan Prado

3.1.5.8.1.5. Administrar marcadores

Tabla III-27: Descripción de Caso de Uso 12: Administrar Marcadores

Nombre:	**Administrar marcadores**		
Identificador:	CU-12		
Actor:	Usuario administrador		
Descripción:	Consiste en un formulario que permite ingresar marcadores de posicionamiento los cuales son referencia para los usuarios con rol Android.		
Precondiciones:	1. Tener cuenta con perfil administrador.		
Post-condiciones:			
Flujo principal:	**ACTOR**		**SISTEMA**
	1 El usuario ingresa a la pantalla principal de la aplicación.	**2**	El sistema mostrará el formulario principal.
	3 El usuario da clic en opción geo posicionamiento.	**4**	El sistema mostrará un mapa con la posición actual del usuario.
	5 El usuario da clic sobre el mapa.	**6**	El sistema muestra un dialogo para ingresar el usuario al que pertenece el marcador
	7 El usuario ingresa el usuario que se presentará a través de un combo box, y da clic en guardar	**8**	El mapa se actualiza mostrando los marcadores que existen actualmente en el mapa.
Flujo alternativo			
Resultado:	Se mostrará los marcadores actuales dentro del mapa.		

Elaborado por: Diego Ponce – Juan Prado

3.1.5.8.2. Administrar Mensajería

3.1.5.8.2.1. Enviar mensaje

Tabla III-28: Descripción de Caso de Uso 13: Enviar Mensaje

Nombre:	**Administrar mensajería – Enviar mensaje**			
Identificador:	CU-13			
Actor:	Usuario administrador			
Descripción:	Consiste en un formulario que permite mostrar y enviar mensajes por usuario.			
Precondiciones:	1. Tener cuenta con perfil administrador. 2. Tener id de servidor de cuenta GCM 3. Tener serial de servidor de GCM			
Post-condiciones:				
Flujo principal:		**ACTOR**		**SISTEMA**
	1	El usuario ingresa a la pantalla principal de la aplicación.	**2**	El sistema mostrará el formulario principal.
	3	El usuario da clic en opción chat.	**4**	El sistema mostrará un formulario con los contactos y mensajes de cada contacto.
	5	El usuario da clic el contacto al que se va a enviar el mensaje.	**6**	El sistema carga los mensajes correspondientes al usuario
	7	El usuario llena el campo de mensaje, el cual se va a enviar al contacto, y da clic en enviar	**8**	El sistema envía el mensaje através de GCM y guarda en la base de datos el mensaje.
			9	La aplicación del contacto devuelve un mensaje ACT, confirmando la recepción.
			10	Actualiza pantalla y muestra mensaje.
Flujo alternativo			**9.1**	Si la aplicación del contacto no devuelve un mensaje ACT el mensaje se guarda con estado pendiente para ser enviado nuevamente.
Resultado:	Se enviará el mensaje al contacto correspondiente.			

Elaborado por: Diego Ponce – Juan Prado

3.1.5.8.2.2. Recibir Mensaje

Tabla III-29: Descripción de Caso de Uso: Recibir Mensaje

Nombre:	**Administrar mensajería – Recibir mensaje**			
Identificador:	CU-14			
Actor:	Usuario administrador			
Descripción:	Consiste en un formulario que permite mostrar y recibir mensajes por usuario.			
Precondiciones:	1. Tener cuenta con perfil administrador. 2. Tener id de servidor de cuenta GCM 3. Tener serial de servidor de GCM			
Post-condiciones:				
Flujo principal:	**ACTOR**		**SISTEMA**	
			1	El sistema recibe mensaje a través de GCM y guarda en la base de datos el mensaje.
			2	Identifica el usuario al que pertenece el mensaje.
			3	Guarda mensaje en base de datos
			4	El sistema devuelve al contacto un mensaje ACT, confirmando la recepción
			5	Notifica al usuario la recepción del mensaje
	6	El usuario da clic en opción chat.	**7**	El sistema mostrará un formulario con los contactos y mensajes de cada contacto.
	8	El usuario da clic sobre el contacto que envió el mensaje	**9**	El sistema carga los mensajes correspondientes al usuario
Flujo alternativo				
Resultado:	Se recibirá y almacenará el mensaje del contacto.			

Elaborado por: Diego Ponce – Juan Prado

3.1.5.8.3. Visualizar Posicionamiento

Tabla III-30: Descripción de Caso de Uso: Visualizar Posicionamiento

Nombre:	**Visualizar Posicionamiento**			
Identificador:	CU-15			
Actor:	Usuario administrador			
Descripción:	Consiste en un formulario que permite mostrar y recibir mensajes por usuario.			
Precondiciones:	1. Tener cuenta con perfil administrador. 2. Tener id de servidor de cuenta GCM 3. Tener serial de servidor de GCM			
Post-condiciones:				
Flujo principal:	**ACTOR**		**SISTEMA**	
			1	Sistema recibe la posición a través de un mensaje de GCM
			2	Identifica de qué usuario pertenece esa posición.
			3	Almacena en base de datos posición
			4	Devuelve mensaje ACT a través de GCM
	5	El usuario ingresa a la pantalla principal de la aplicación.	**6**	El sistema mostrará el formulario principal.
	7	El usuario da clic en opción geo localización.	**8**	El sistema mostrará un formulario con el mapa de Google Maps
			9	El sistema carga las posiciones de los usuarios
Flujo alternativo			**9.1**	Si el usuario no se encuentra en la pantalla de geo localización las posiciones de los contactos se almacenan en la base de datos en background.
Resultado:	Se recibirá y almacenará las posiciones de los contactos.			

Elaborado por: Diego Ponce – Juan Prado

3.1.5.9. Diagramas de actividad: Monitorización Virtual Móvil

3.1.5.9.1. Registrar Usuario

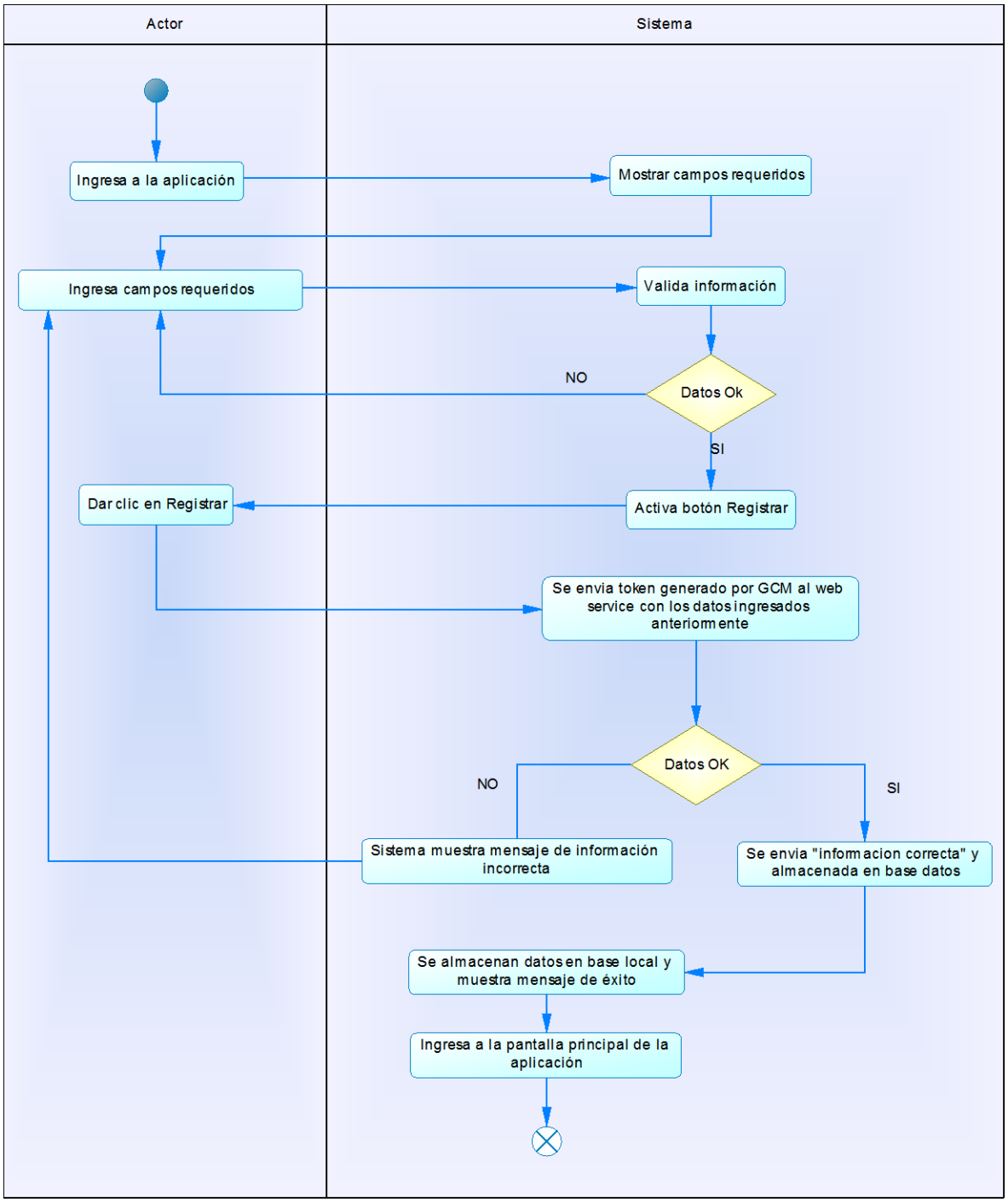


Diagrama III-4: Diagrama de Actividades 1: Registro de Usuario

Elaborado por: Diego Ponce – Juan Prado

3.1.5.9.2. Actualizar Contactos

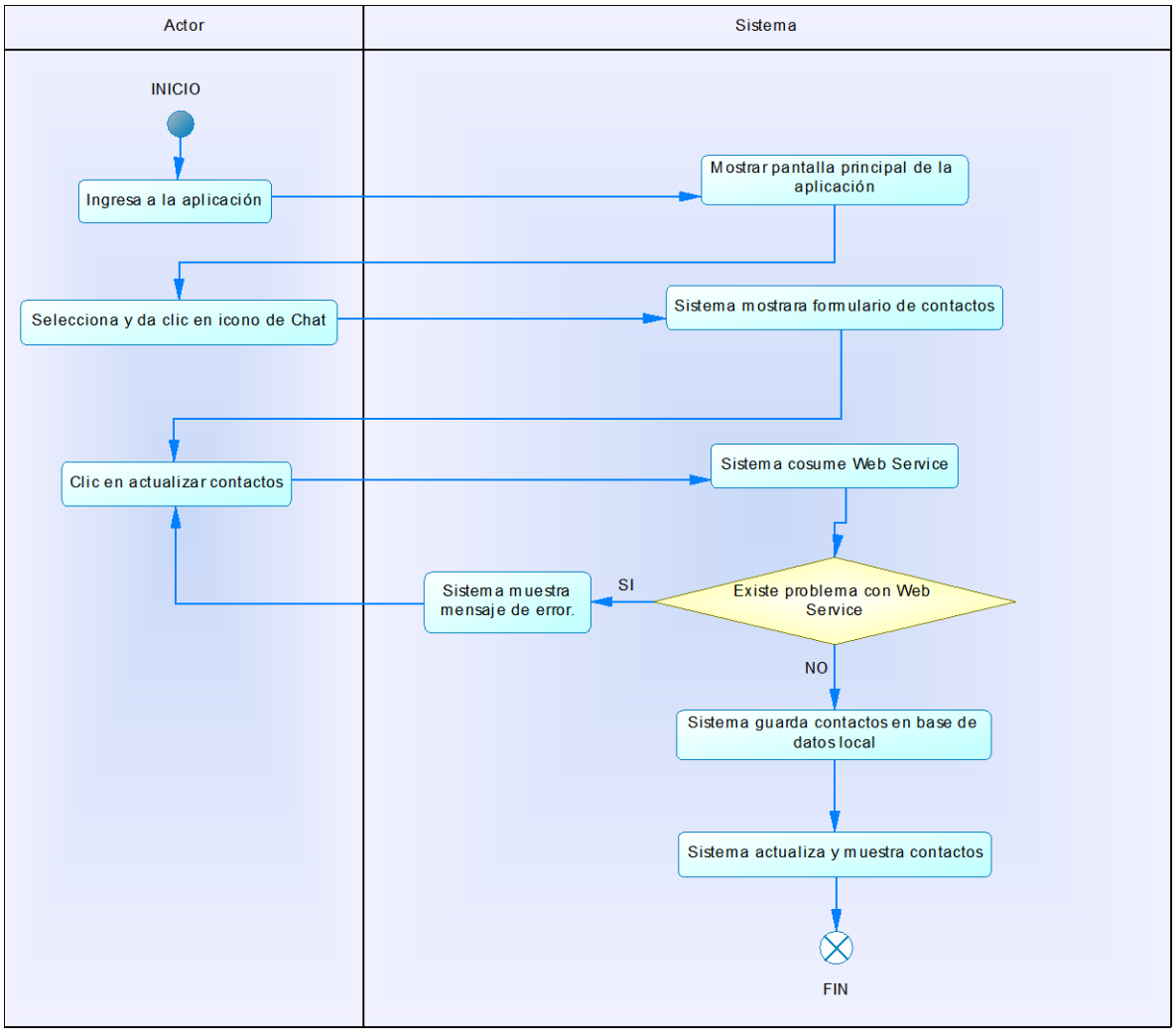


Diagrama III-5: Diagrama de Actividades: Actualización de Usuario

Elaborado por: Diego Ponce – Juan Prado

3.1.5.9.3. Administrar Mensajes

3.1.5.9.3.1. Enviar mensajes

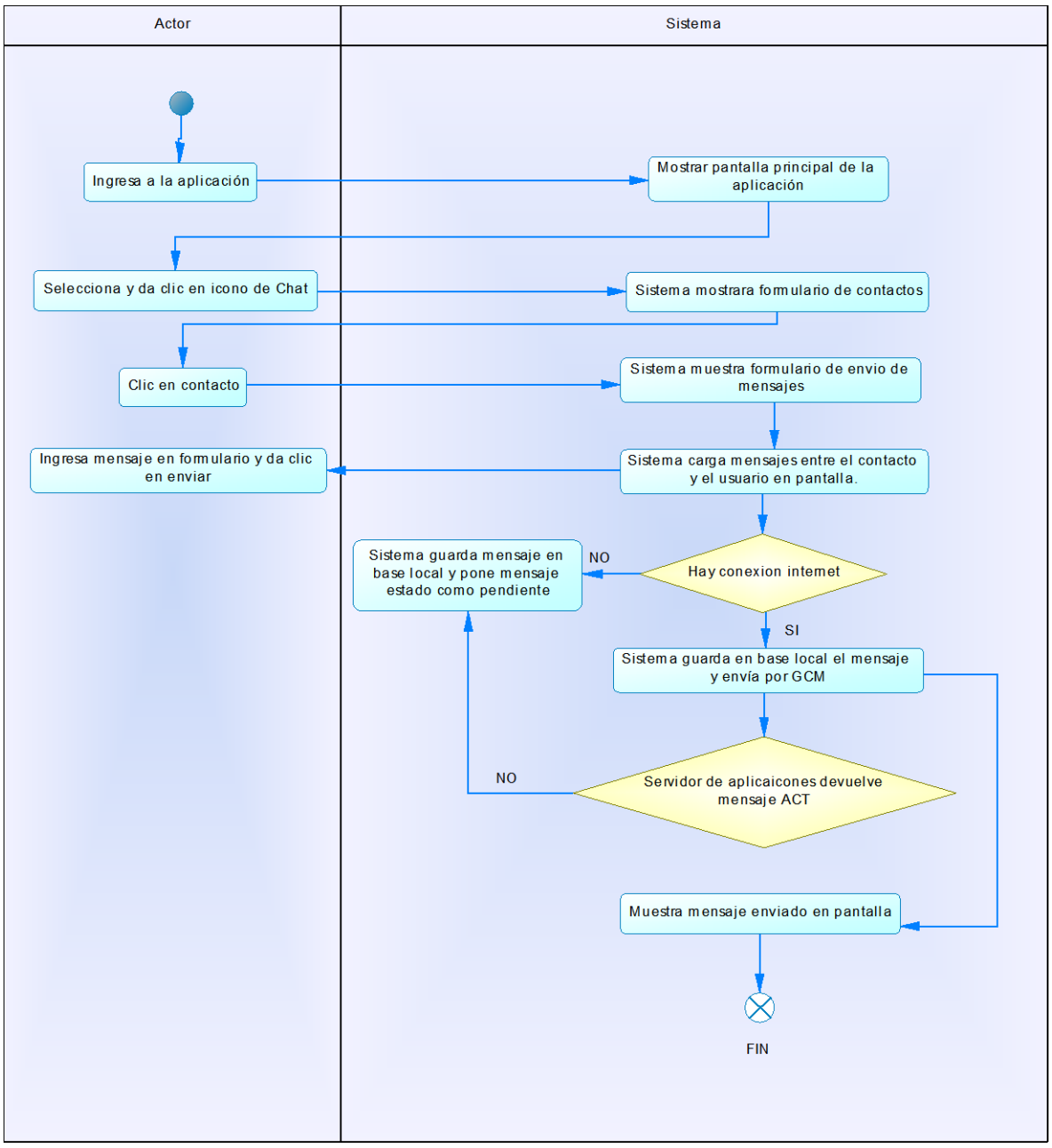


Diagrama III-6: Diagrama de Actividades 3: Envío de Mensajes

Elaborado por: Diego Ponce – Juan Prado

3.1.5.9.3.2. Recibir mensajes

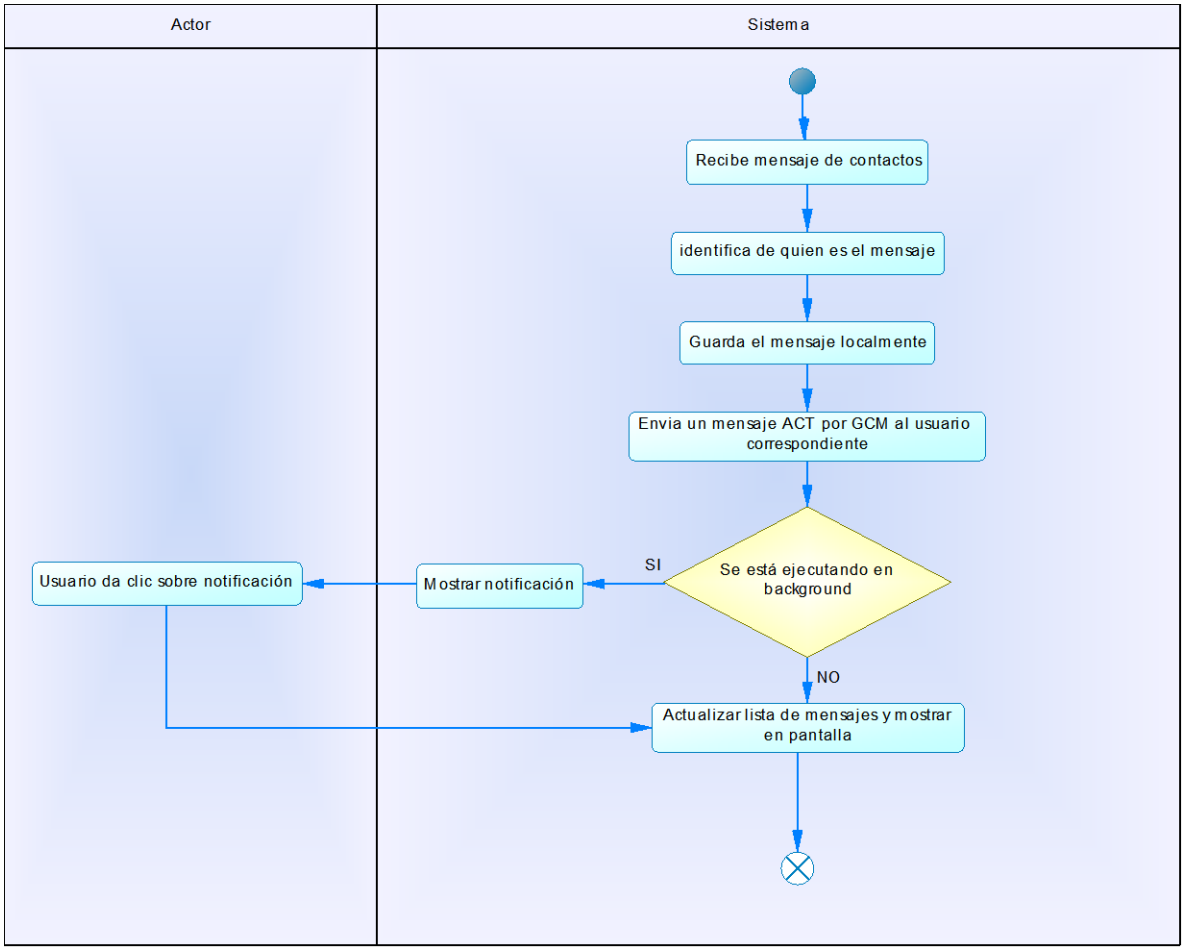


Diagrama III-7: Diagrama de Actividades 4: Recepción de Mensajes

Elaborado por: Diego Ponce – Juan Prado

3.1.5.9.4. Administrar Marcadores

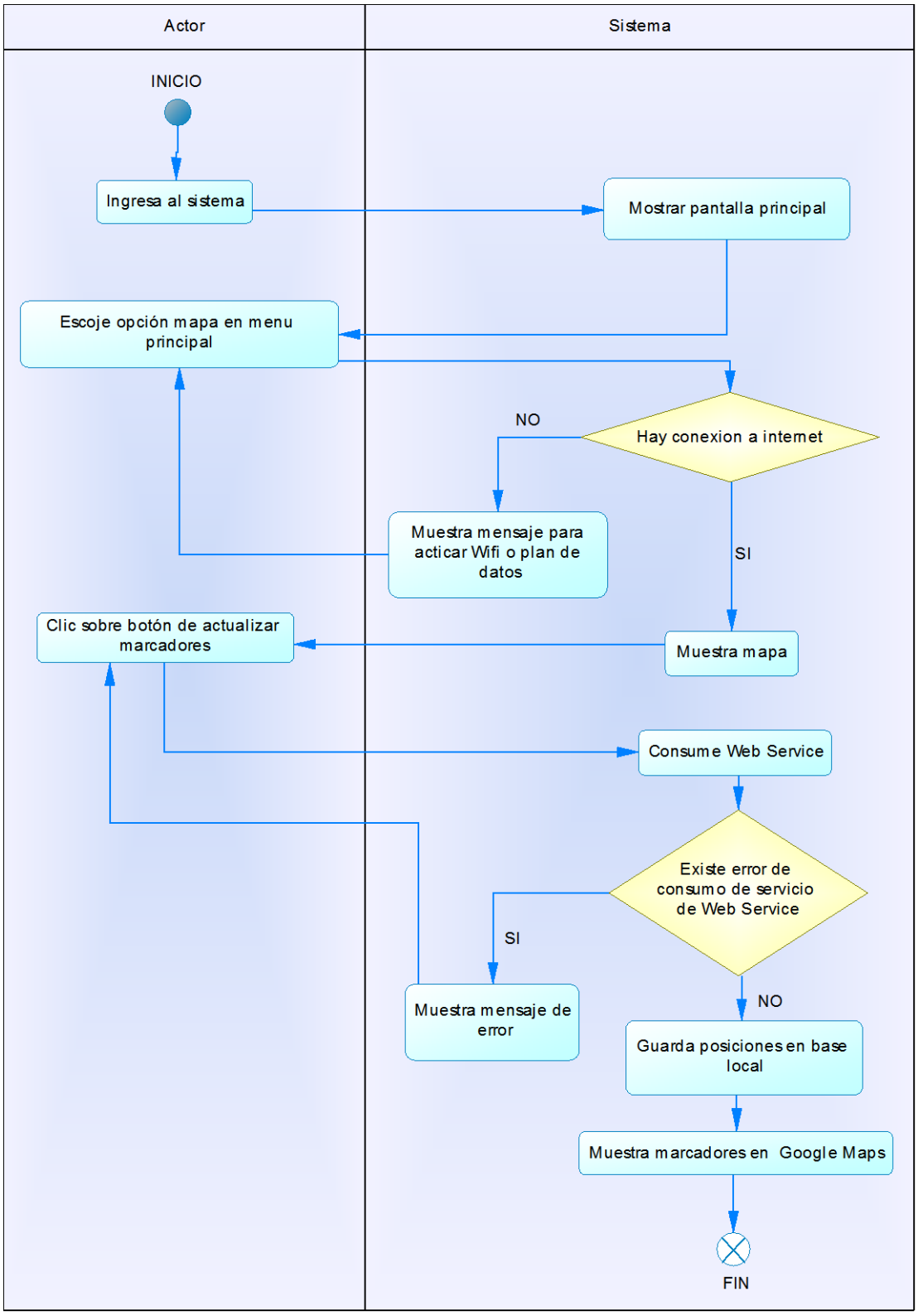


Diagrama III-8: Diagrama de Actividades 5: Administración de Marcadores

Elaborado por: Diego Ponce – Juan Prado

3.1.5.9.5. Administrar Posición

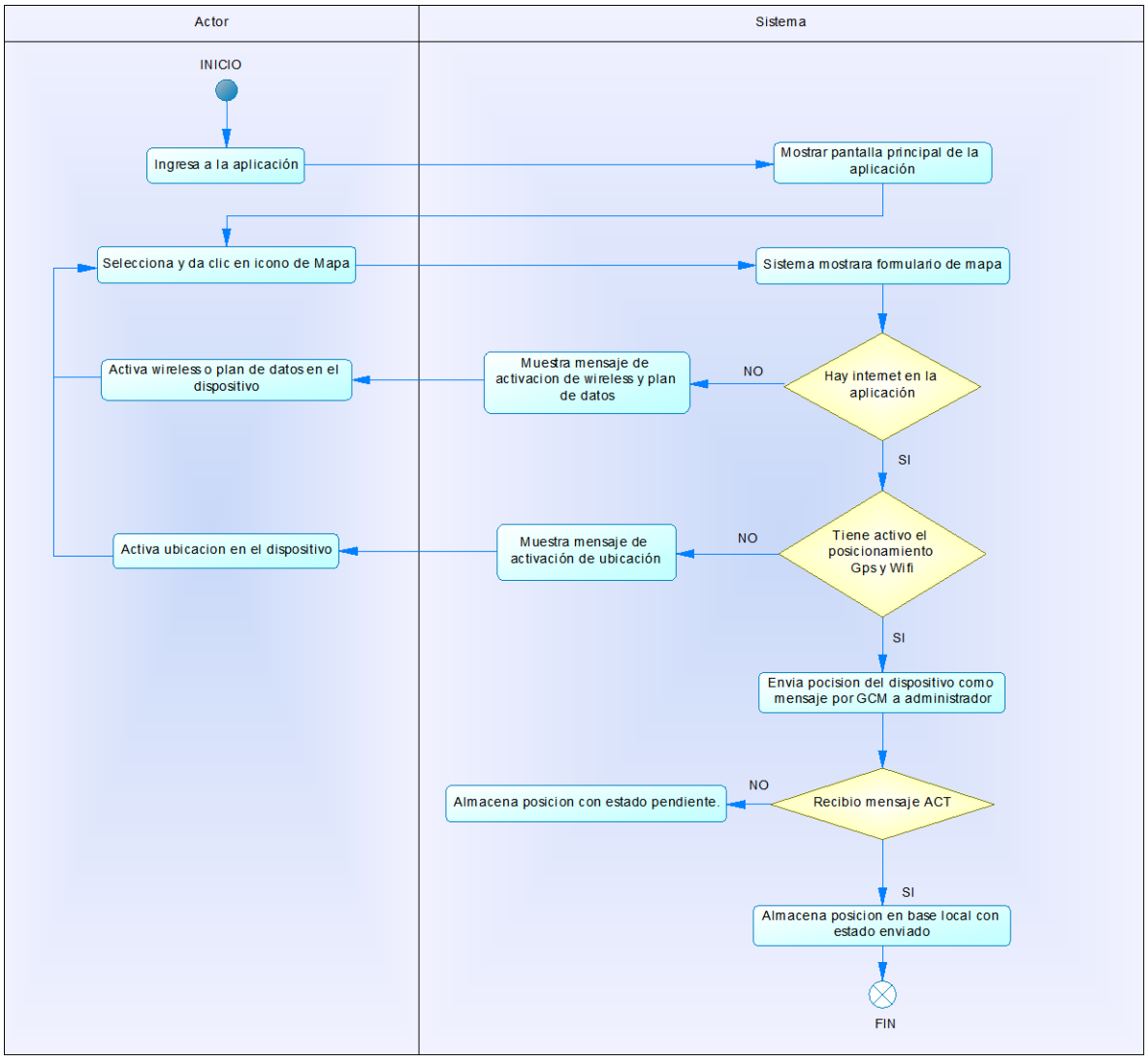


Diagrama III-9: Diagrama de Actividades 6: Administración de la Posición

Elaborado por: Diego Ponce – Juan Prado

3.1.5.10. Diagramas de actividad: Monitorización Virtual Administración Web

3.1.5.10.1. Administrar Usuarios

3.1.5.10.1.1. Ingresar Usuario

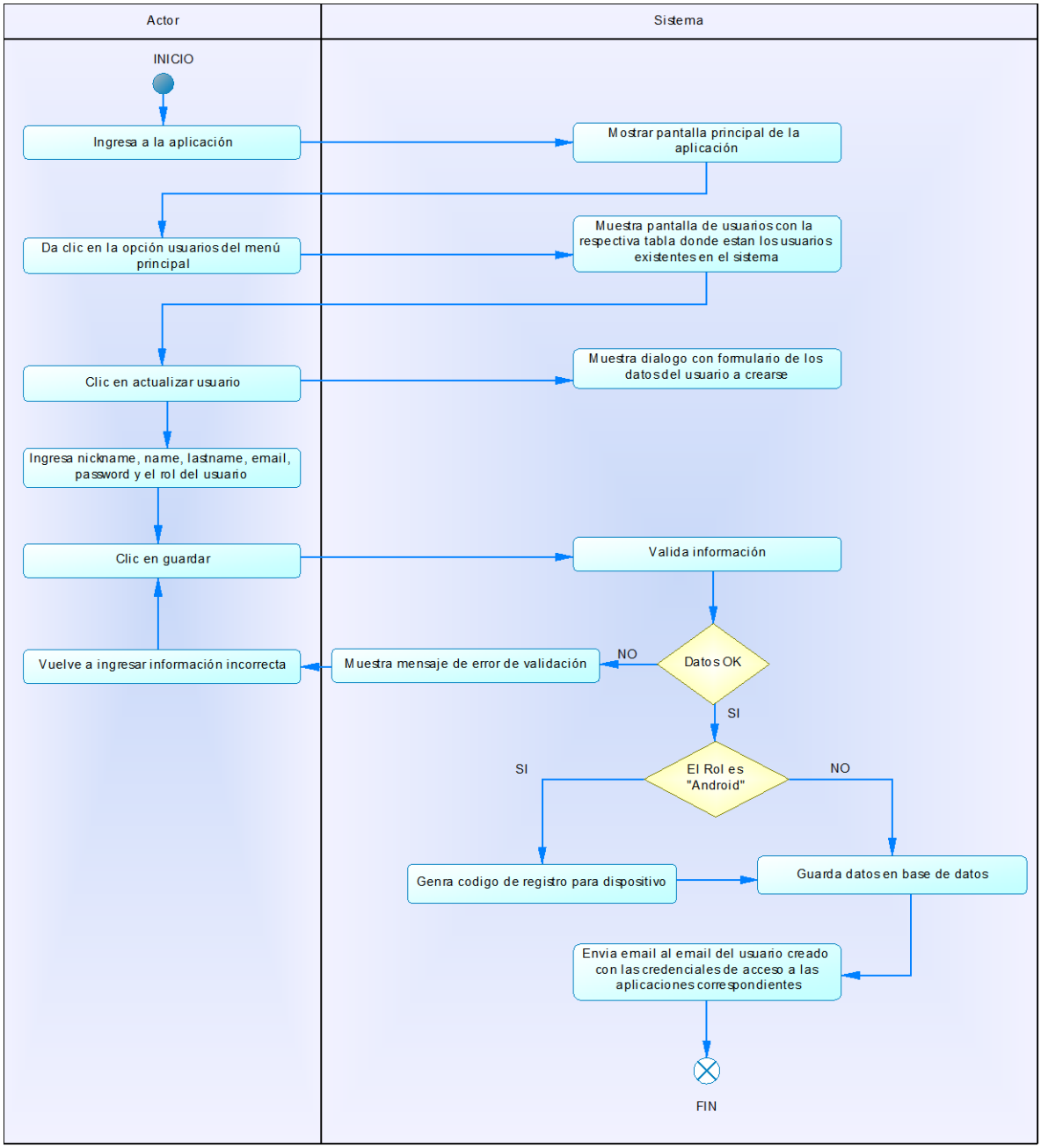


Diagrama III-10: Diagrama de Actividades 7: Ingreso de Usuario

Elaborado por: Diego Ponce – Juan Prado

3.1.5.10.1.2. Modificar Usuario

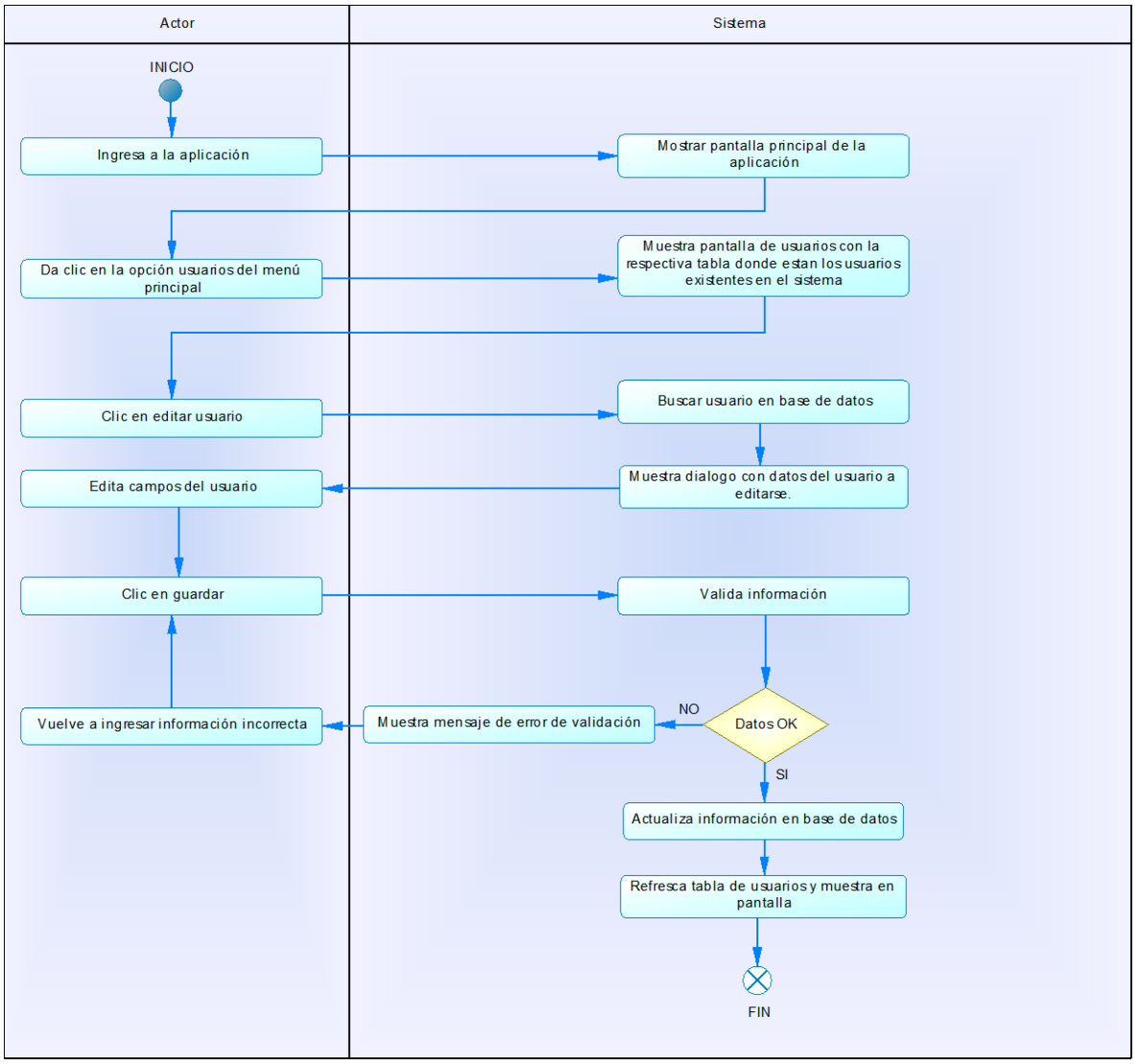


Diagrama III-11: Diagrama de Actividades 8: Modificación de Usuario

Elaborado por: Diego Ponce – Juan Prado

3.1.5.10.1.3. Eliminar Usuario

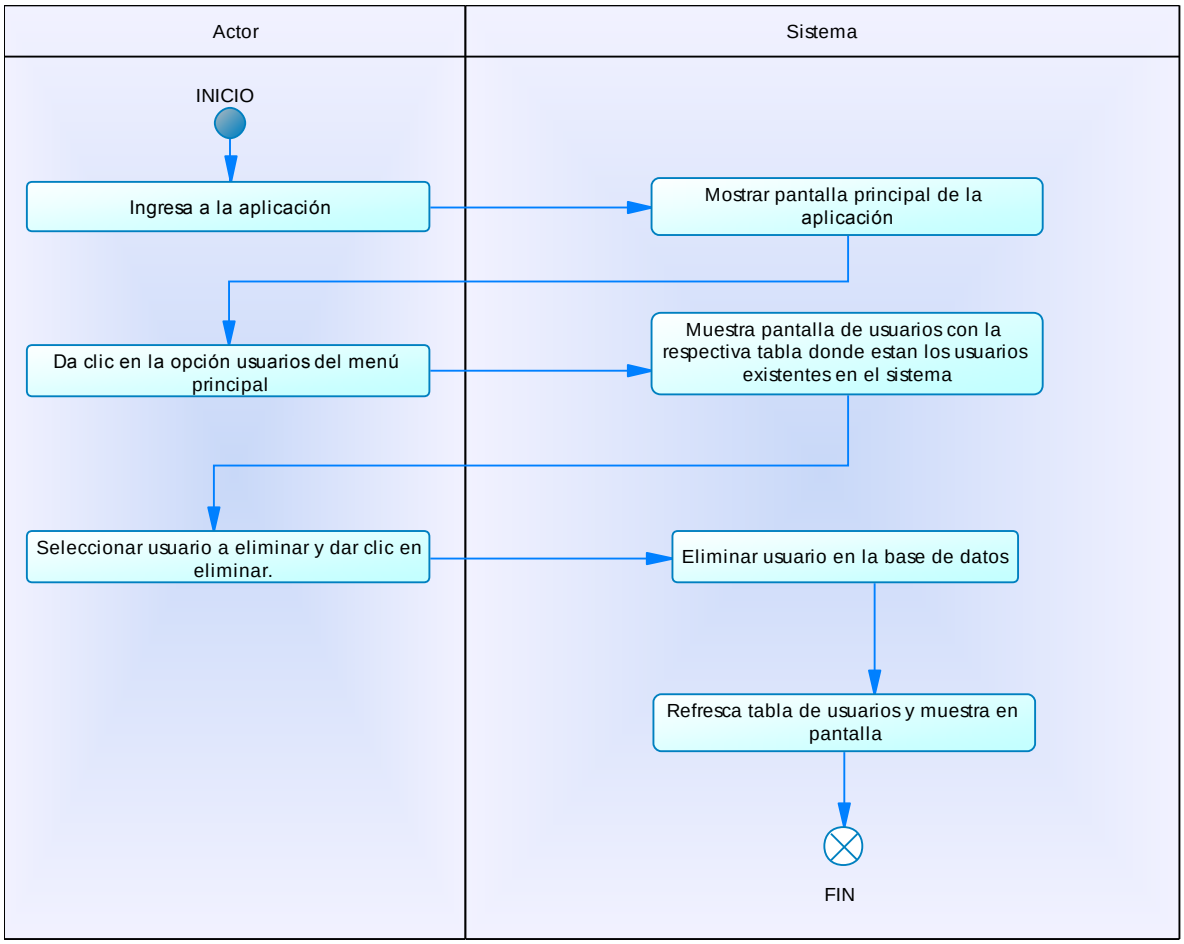


Diagrama III-12: Diagrama de Actividades 9: Eliminación de Usuario

Elaborado por: Diego Ponce – Juan Prado

3.1.5.10.1.4. Consultar Usuario

3.1.5.10.1.4.1. Consulta General

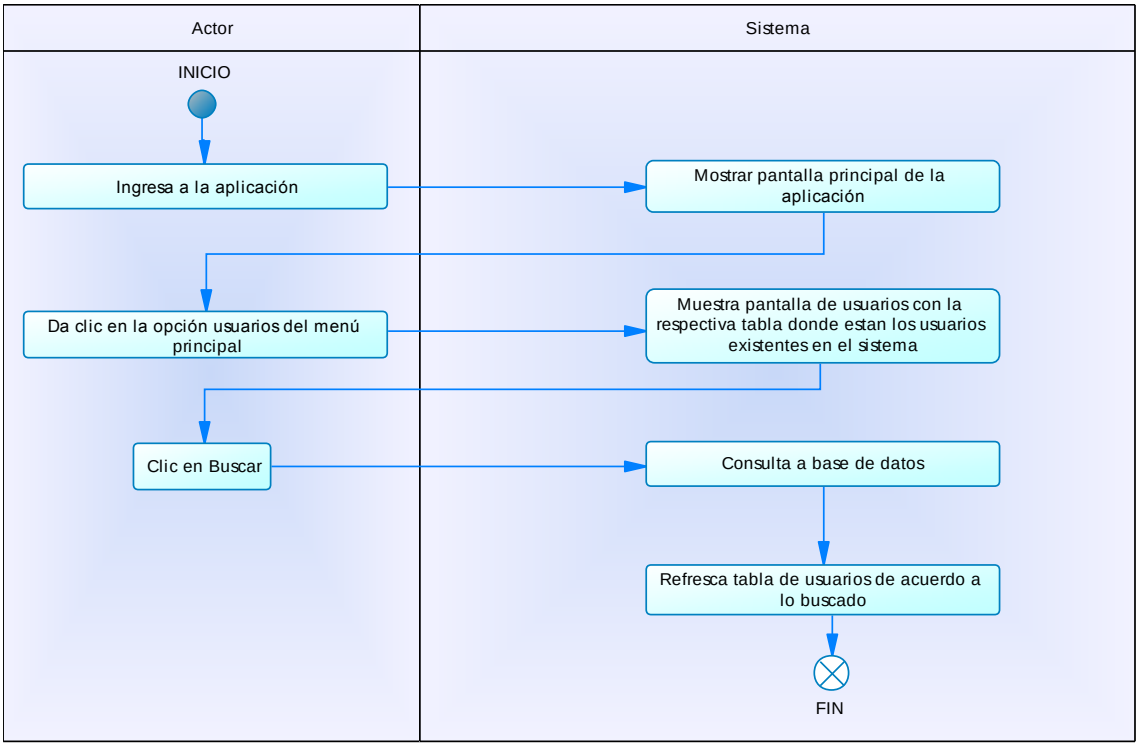


Diagrama III-13: Diagrama de Actividades 10: Consulta de Usuario – Consulta General

Elaborado por: Diego Ponce – Juan Prado

3.1.5.10.1.4.2. Consulta por parámetro

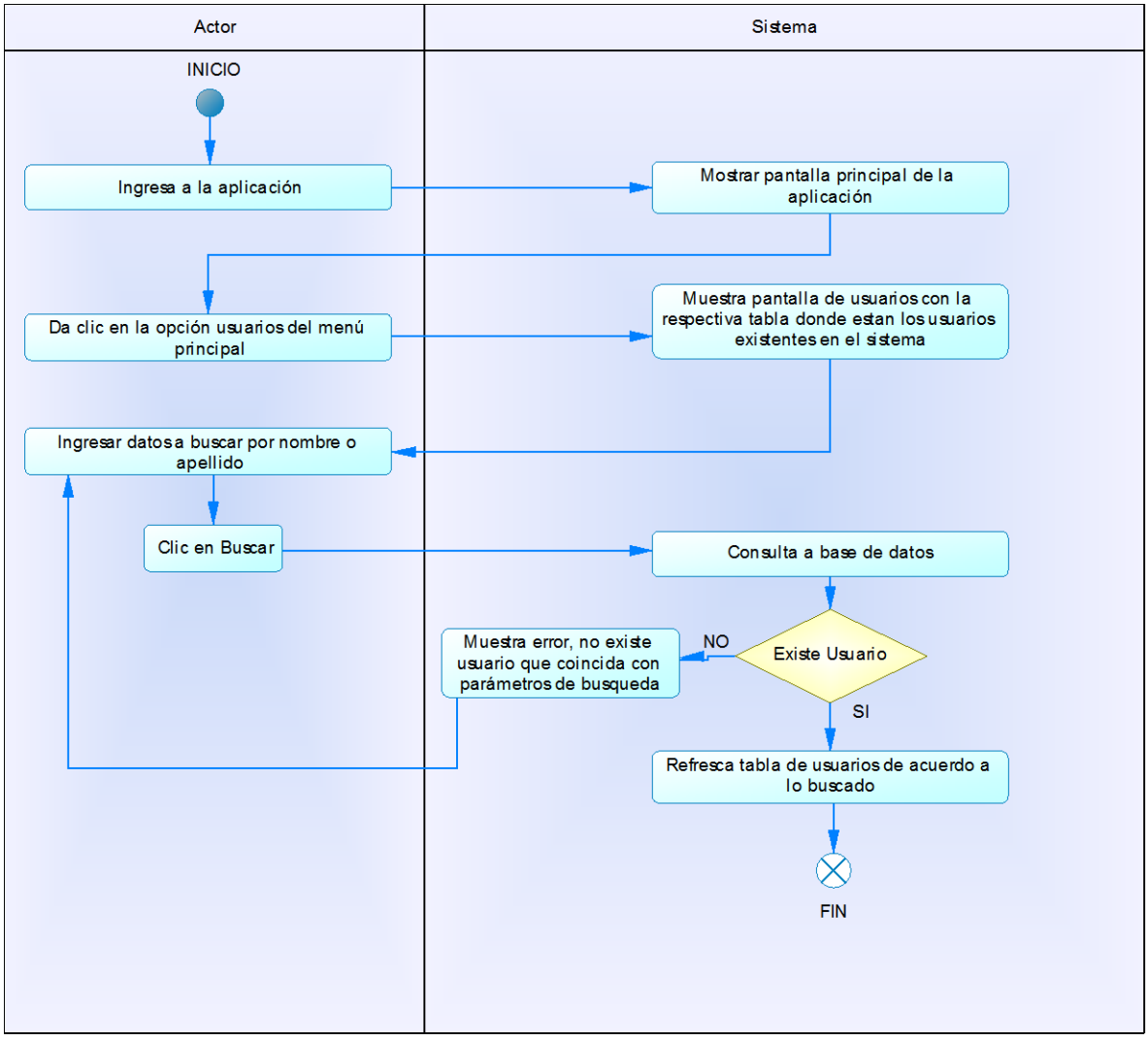


Diagrama III-14: Diagrama de Actividades 11: Consulta de Usuario – Consulta de Parámetro

Elaborado por: Diego Ponce – Juan Prado

3.1.5.10.2. Administrar Marcadores

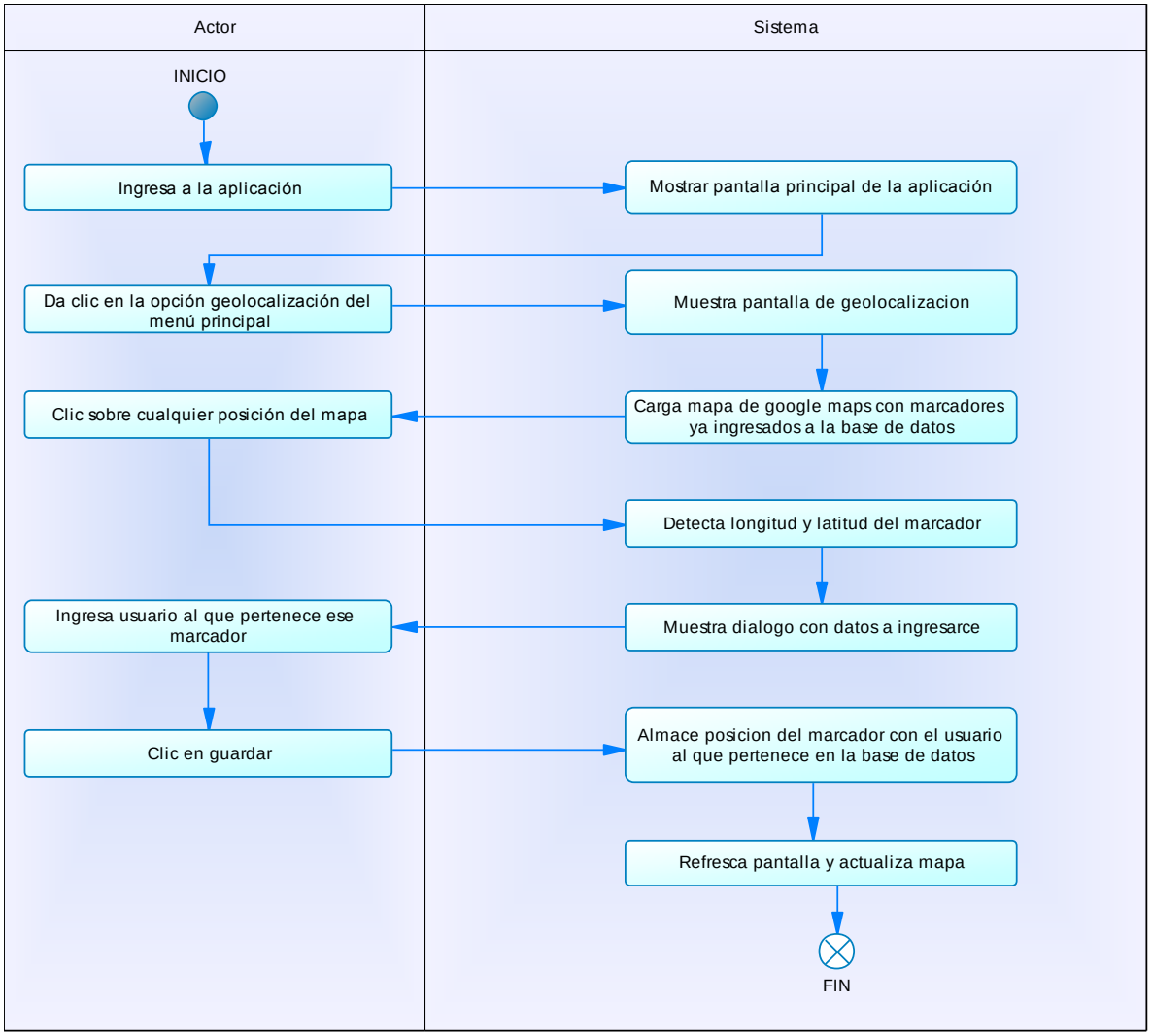


Diagrama III-15: Diagrama de Actividades 12: Administración de Marcadores

Elaborado por: Diego Ponce – Juan Prado

3.1.5.10.3. Administrar Mensajería

3.1.5.10.3.1. Enviar Mensaje

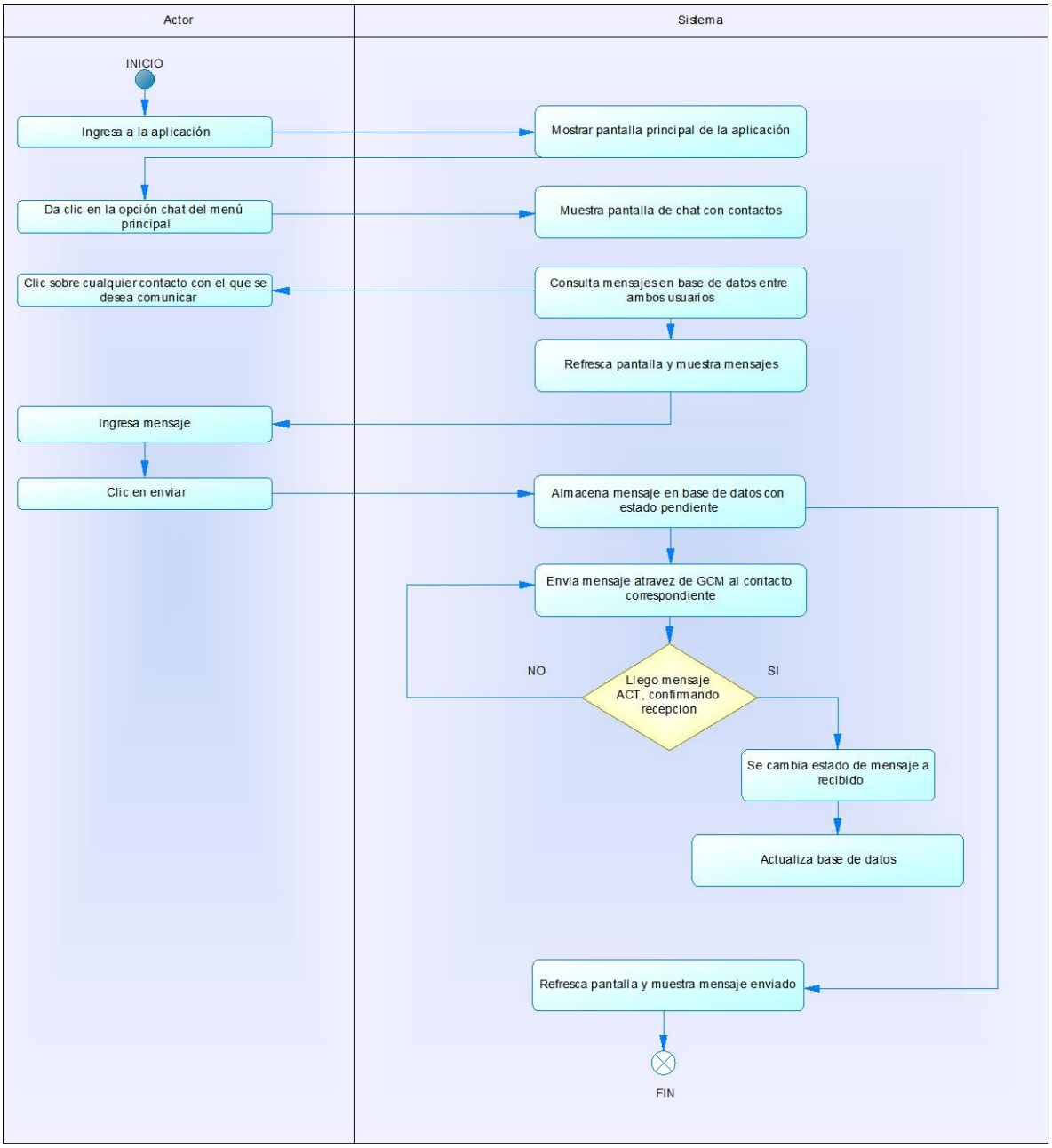


Diagrama III-16: Diagrama de Actividades 13: Mensajería – Envío de Mensajes

Elaborado por: Diego Ponce – Juan Prado

3.1.5.10.3.2. Recibir Mensaje

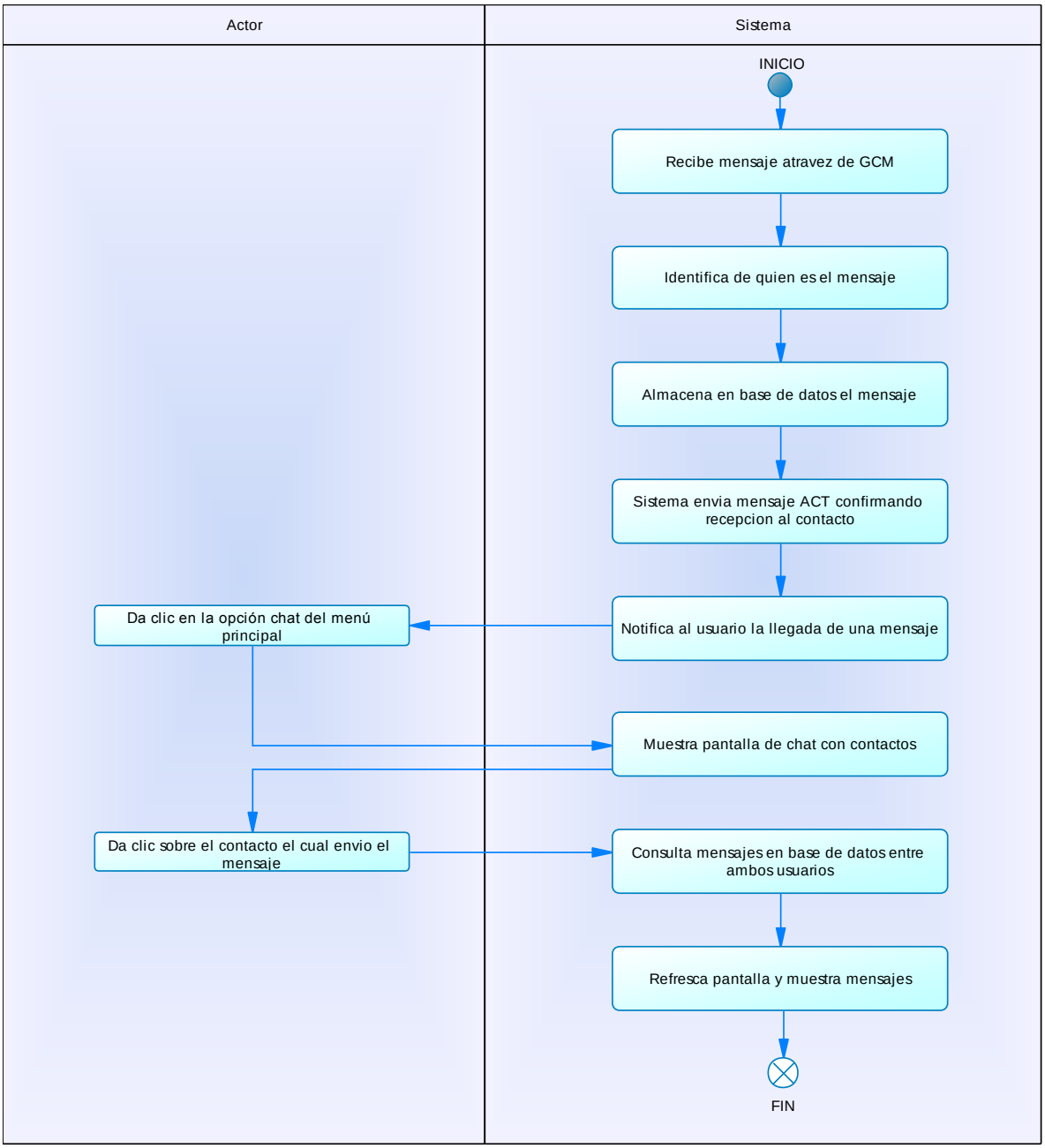


Diagrama III-17: Diagrama de Actividades 14: Mensajería – Recepción de Mensajes

Elaborado por: Diego Ponce – Juan Prado

3.1.5.10.4. Visualizar Posicionamiento

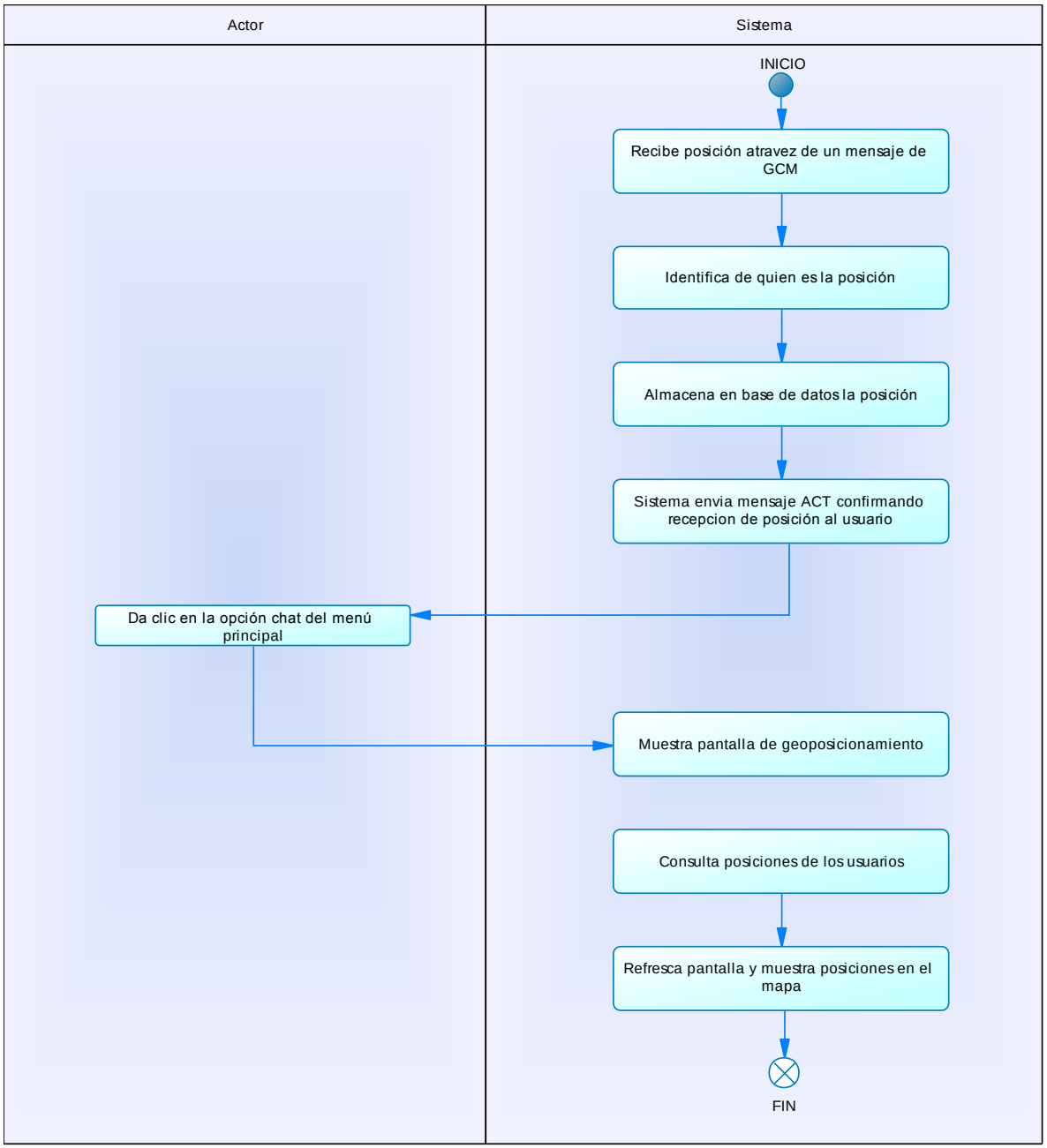


Diagrama III-18: Diagrama de Actividades 15: Visualización de Posición

Elaborado por: Diego Ponce – Juan Prado

3.1.6. DIAGRAMA CONCEPTUAL

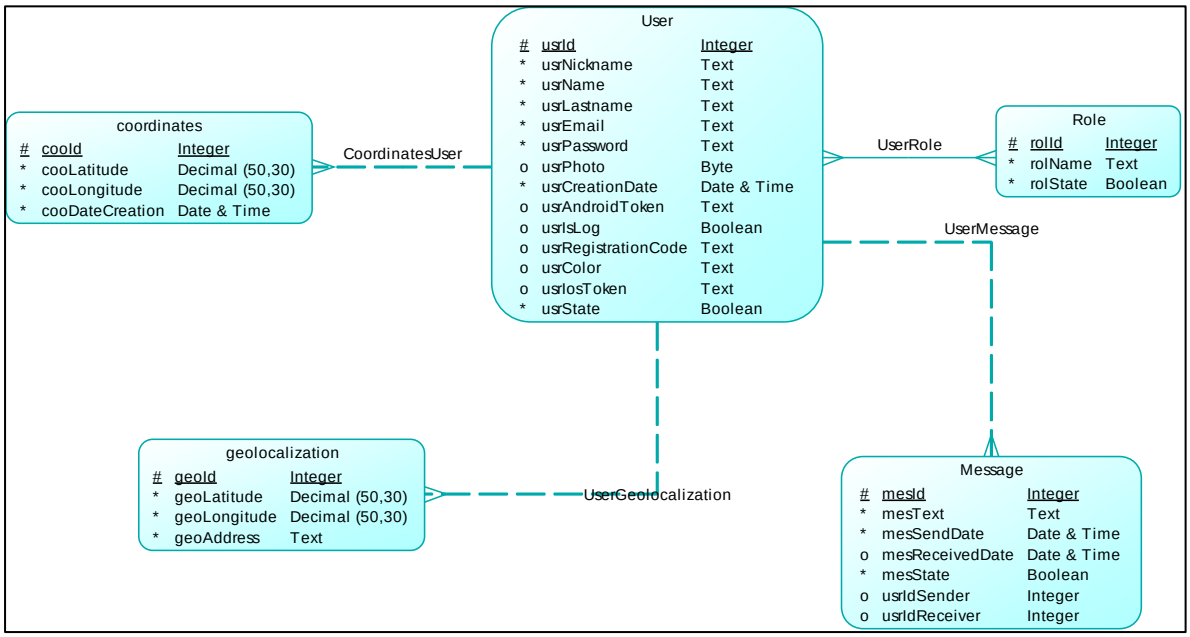


Diagrama III-19: Diagrama Conceptual de Proyecto de Desarrollo de Software

Elaborado por: Diego Ponce – Juan Prado

3.1.7. DIAGRAMA DE CLASES

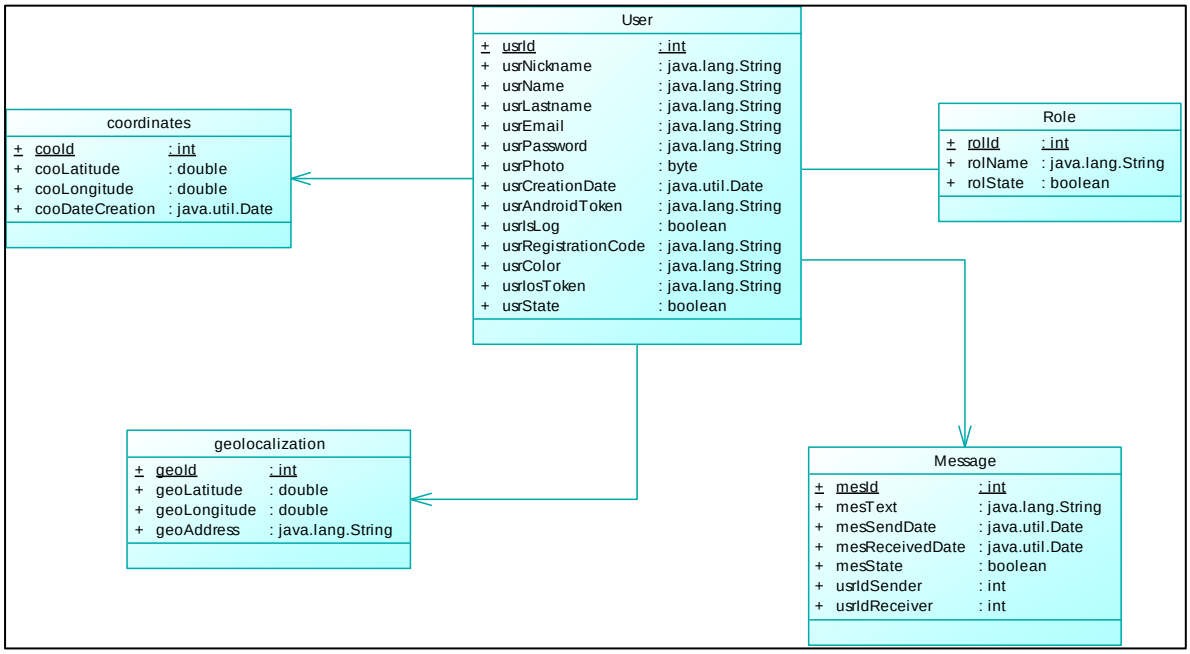


Diagrama III-20: Diagrama de Clases para el Proyecto de Desarrollo de Software

Elaborado por: Diego Ponce – Juan Prado

CAPÍTULO IV.

DISEÑO DEL PROTOTIPO

Dentro del proyecto de software tanto móvil como en la plataforma web se ha definido estándares de diseño e implementación con respecto a las interfaces del usuario, para esto se ha generado plantillas prototipo, el cual será de apoyo al momento del desarrollo tanto los componentes swing de la aplicación web, como los widgets de la parte móvil del proyecto de desarrollo.

A lo largo del capítulo se irá definiendo los factores a tomar en cuenta para diseñar las respectivas interfaces de usuario, con el objetivo de brindar un ambiente cómodo, intuitivo, y de fácil uso para el usuario final.

4.1. Diseño de Interfaces

"La interfaz gráfica de usuario son todos los elementos gráficos que nos ayudan a comunicarnos con un sistema." (González, 2004).

Dentro del desarrollo para el proyecto de monitoreo y localización de dispositivos móviles se ha definido las interfaces de usuario un punto crítico por el cual se toma en cuenta los siguientes elementos para el diseño de las mismas.

- Usabilidad
- Navegabilidad
- Asistencia
- Intuitiva

- Amigabilidad a los usuarios

El diseño o interfaz que interactúan tanto con el motor de la aplicación en la plataforma web y a la vez, con la aplicación móvil tendrá la responsabilidad de mostrar información puntual solicitada al usuario, la suficiente para su rol y permisos, y con la opción a realizar las acciones de un CRUD al mismo, sea el caso necesario.

Se toma en cuenta que las pantallas de usuario deben ser simples y de carga fácil, para lo cual se considera en la plantilla sectores o áreas predeterminadas que podrán ser reusables por el cache de los navegadores o sistemas operativos incluidos en la gestión de recursos para las aplicaciones.

- Cabeceras de aplicación o sitio web.
- Barra de Menús y herramientas.
- Barra de Búsqueda.
- Áreas de contenido de información.
- Tablas de Información.
- Tablero de acciones adicionales.

4.1.1. Diseño de Interfaces para la aplicación web administrativa de monitoreo y ubicación de dispositivos móviles

En la aplicación de administrador web se encuentra todas las acciones CRUD y monitoreo de dispositivos móviles que pueda controlar los usuarios con tales permisos y privilegios, por ello se ha creado plantillas de diseño para cada módulo y su respectiva distribución de los objetos o componentes de la aplicación, los cuales son detallados a continuación.

1. Pantalla de Inicio de Sesión

Ilustración IV-1: Pantalla Web 1: Sesión de Usuario Administrador

Elaborado por: Diego Ponce – Juan Prado

Esta interfaz está compuesta por dos componentes, la imagen en la parte superior en donde se encuentra el logotipo ya sea de la aplicación o de la empresa a la que está vinculada el servicio de la misma, y un formulario para la entrada de parámetros del usuario la cual previamente ha sido proporcionada.

2. Plantilla Principal para módulos de Aplicación Web

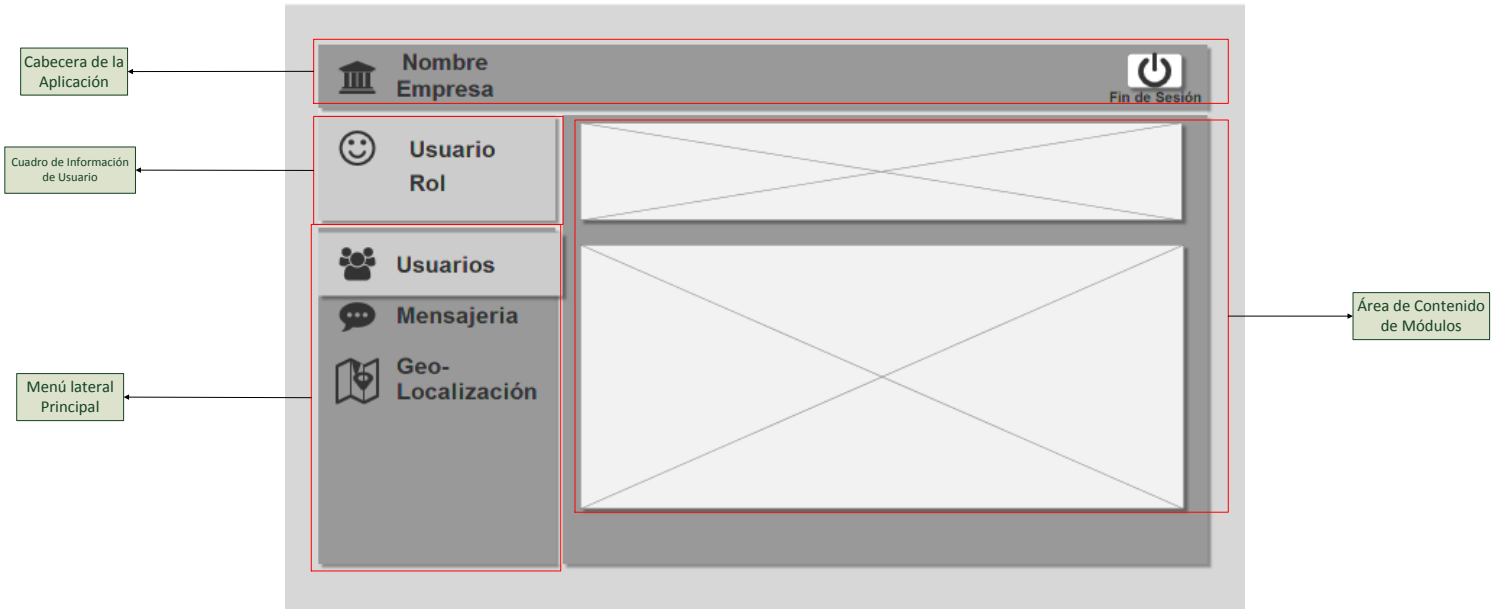


Ilustración IV-2: Plantilla para página principal del aplicativo Web

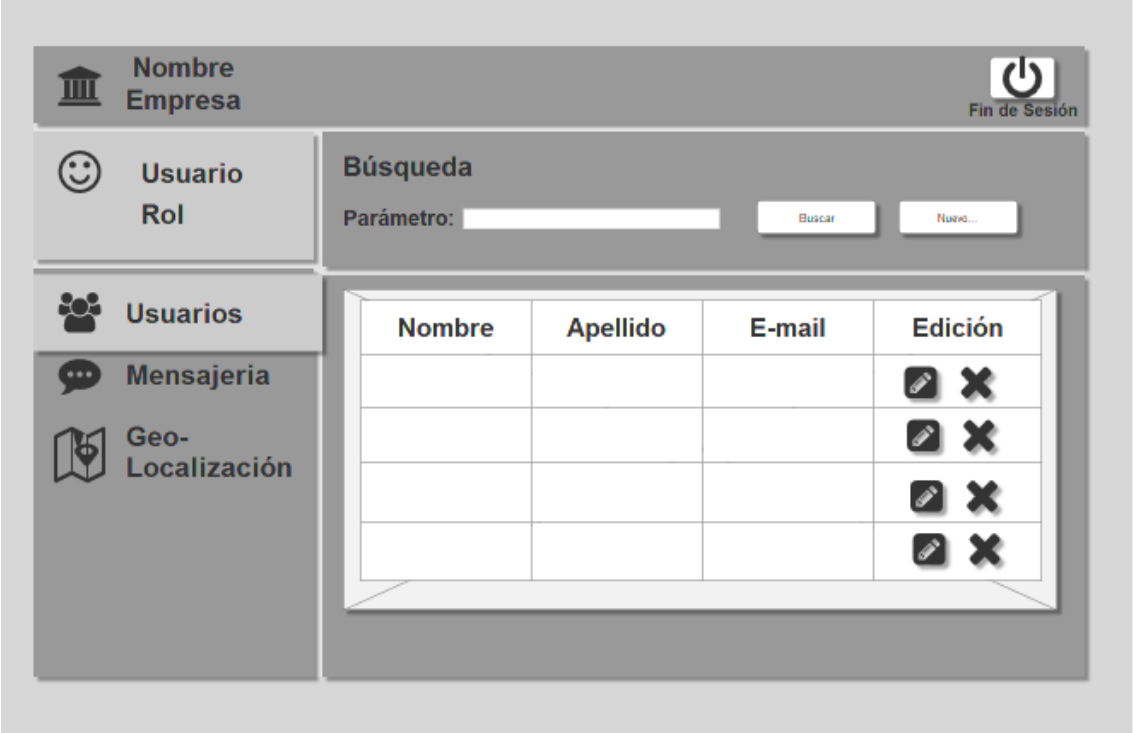
Elaborado por: Diego Ponce – Juan Prado

Esta interfaz es tomada como plantilla principal y será re utilizada para todos los módulos que conformen la aplicación web, la misma que está clasificado en las siguientes secciones:

- **Cabecera:** Se encuentra el nombre ya sea de la aplicación o de la empresa asociada el servicio de la misma, adicional se encuentra un botón para la finalización de la sesión del usuario administrador.

- **Cuadro de Información del Usuario:** Llevará información corta y multimedia del usuario que ha iniciado sesión dentro del aplicativo.

- **Menú Principal:** Contiene enlaces directos a las interfaces de cada módulo desarrollado la cual se desplegarán en el área de contenido
- **Área de Contenido de módulos:** Aquí se presentarán los componentes



requeridos para reflejar la funcionalidad de los módulos correspondientes a la aplicación web y para que el usuario pueda interactuar con ellos.

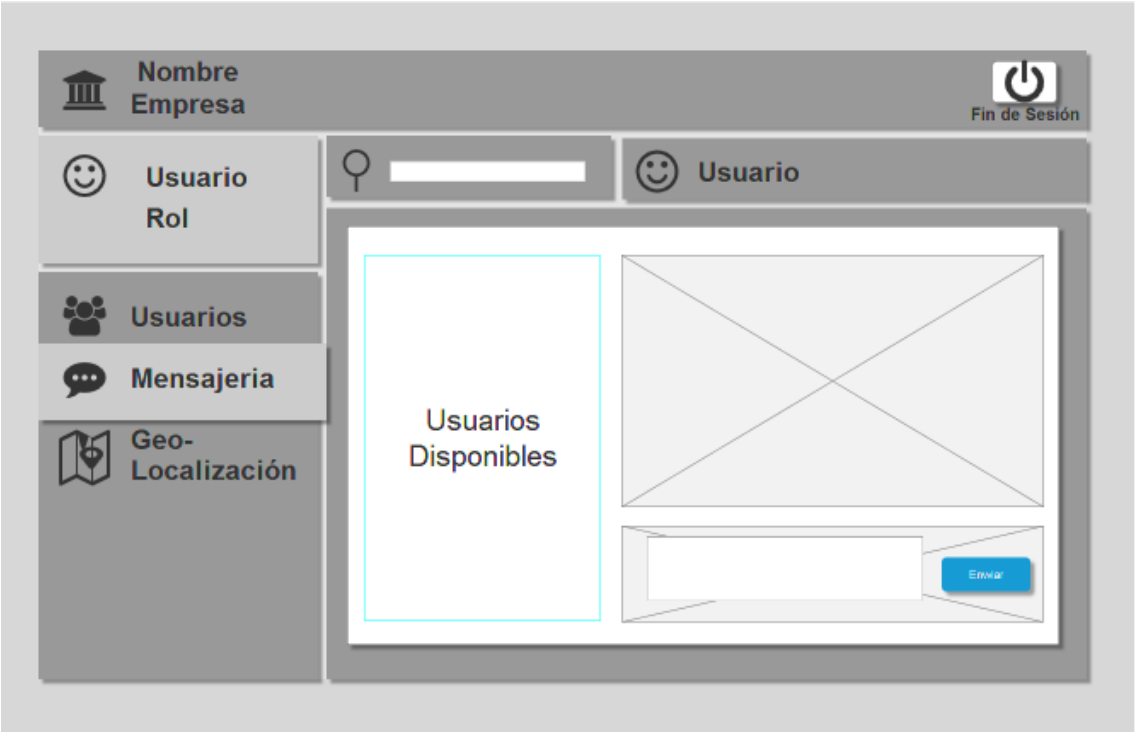
3. Pantalla de Gestión de Usuarios

Ilustración IV-3: Pantalla Web 2: Gestión de Usuarios

Elaborado por: Diego Ponce – Juan Prado

En el área de contenido para el módulo de gestión de usuarios se conforma principalmente por una tabla de reporte de datos en la cual aparecerá información corta de los usuarios registrados en la aplicación como se muestra en cada columna, adicional se podrá realizar operaciones de CRUD en la última de ellas como se puede apreciar en los íconos de la columna “Edición”, el primer ícono para modificar un registro o usuario sea el caso, y el segundo para eliminarlo.

También se toma en cuenta el diseño de un área de búsqueda e ingreso de nuevo usuario situado en la parte superior de la tabla de reporte de datos.



4. Pantalla de Mensajería Instantánea

Ilustración IV-4: Pantalla Web 3: Mensajería Instantánea – Chat

Elaborado por: Diego Ponce – Juan Prado

Dentro del área de contenido para el módulo de mensajería se encuentra componentes idóneos para un pequeño sistema de chat, en el cual tendrá una lista lateral donde estarán precargados los usuarios disponibles a un enlace de conversación digital y en el lado izquierdo el área donde se visualizarán los mensajes tanto enviados como recibidos.

En la parte inferior del área de mensajes procesados está un pequeño componente para enviar mensajes a usuarios previamente seleccionados en la lista anteriormente mencionada.

Finalmente, en la parte superior de la lista de usuarios se encuentra una pequeña herramienta para buscar un usuario predeterminado.

5. Pantalla de Geo-Localización

Ilustración IV-5: Pantalla Web 4: Geo-Localización

Elaborado por: Diego Ponce – Juan Prado

En esta interfaz se implementará un componente API para la visualización de mapas digitales en los cuales se podrá revisar el sitio donde se encuentran los dispositivos móviles registrados, así como sus trayectorias.

4.1.2. Diseño de Interfaces para la aplicación móvil de monitoreo y ubicación de dispositivos móviles

La aplicación móvil para el proyecto de desarrollo se ha clasificado en tres componentes o plantillas principales en las cuales se denotan características de navegabilidad y comodidad

visual para el usuario, por este motivo se ha distribuido las interfaces en forma de fragmentos y actividades de la siguiente manera.

1. **Pantalla de inicio de sesión**

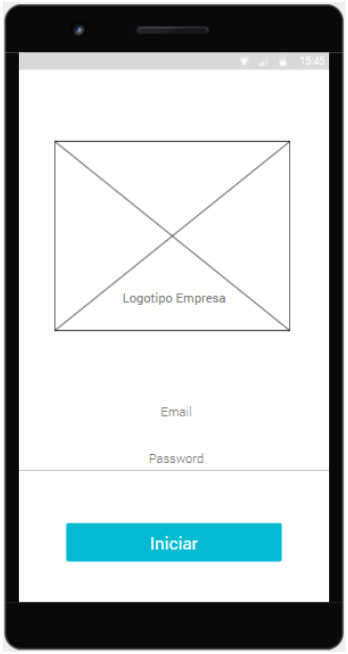


Ilustración IV-6: Pantalla Móvil 1: Inicio de Sesión

Elaborado por: Diego Ponce – Juan Prado

La Pantalla de inicio de sesión se encuentra de manera similar al aplicativo web, una imagen en la parte superior la cual reflejará ya sea el logotipo de la aplicación o de la empresa a la cual está asociada el servicio de la misma y un formulario en la parte inferior donde se ingresará información del usuario necesaria para ingresar a la aplicación cliente.

2. Menú Principal – Sección de Fragmentos

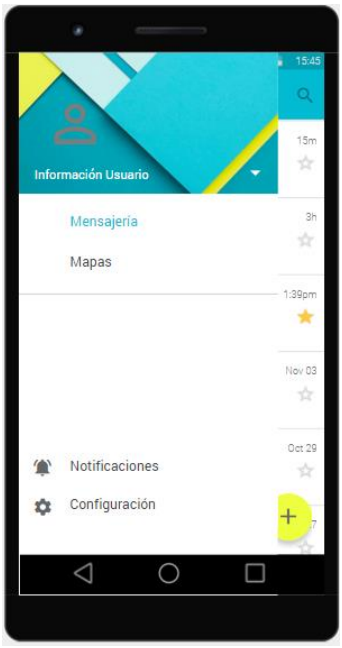


Ilustración IV-7: Pantalla Móvil 2: Fragmento de Menú Principal

Elaborado por: Diego Ponce – Juan Prado

Este fragmento está conformado principalmente por los enlaces a los módulos funcionales que conforman la aplicación cliente inicial, cada uno de ellos mostrará un fragmento diferente con sus respectivos componentes para lograr una adecuada interacción con el usuario.

En la parte superior del menú principal se encuentra información corta del usuario con su respectiva multimedia o avatar para poder identificarlo.

3. Pantalla de Contactos

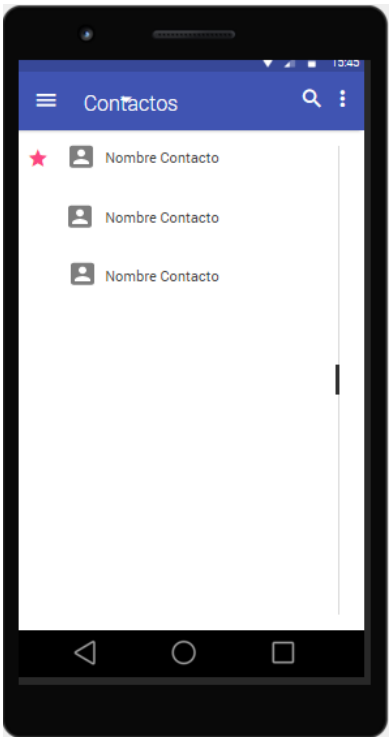


Ilustración IV-8: Pantalla Móvil 3: Fragmento de Contactos

Elaborado por: Diego Ponce – Juan Prado

En esta interface se mostrarán todos los contactos disponibles para una conversación con el usuario ya sea por su estado de conexión, grupo o filtrado según el administrador del aplicativo. Adicionalmente incorpora una pequeña herramienta en la parte superior para la búsqueda de usuarios predeterminados tomando como parámetro su nombre corto.

4. Pantalla de Mensajería instantánea

Ilustración IV-9: Pantalla Móvil 3: Fragmento de Mensajería Instantánea - Chat

Elaborado por: Diego Ponce – Juan Prado

Contiene una interfaz apropiada para conectarse con el motor de chat e interactuar tanto con el servidor de aplicaciones como con otros dispositivos clientes. Cuenta con la información del usuario a enviar los mensajes en la parte superior, así como una pequeña herramienta para buscar usuarios adicionales.

Finalmente incorpora una pequeña barra en la parte inferior con los componentes necesarios para enviar mensajes al usuario previamente seleccionado.

5. Pantalla de Geo-Posicionamiento

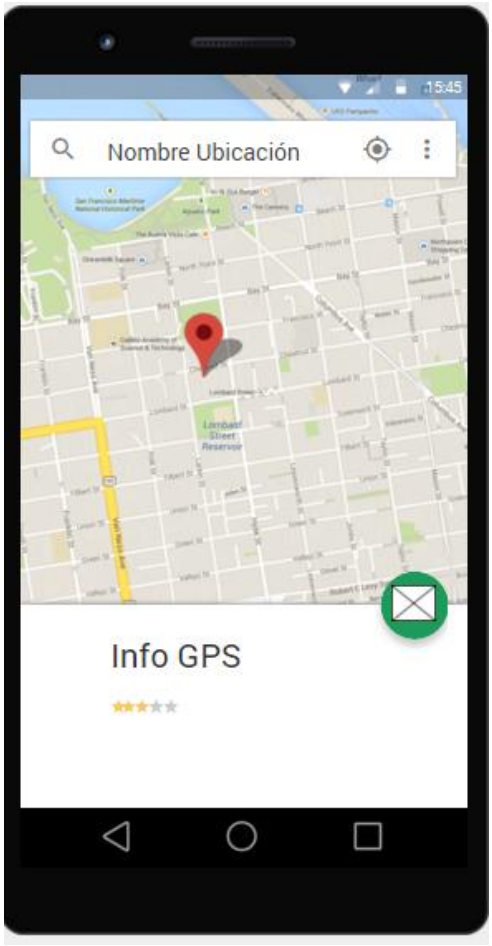


Ilustración IV-10: Pantalla Móvil 4: Fragmento de Mapas Digitales

Elaborado por: Diego Ponce – Juan Prado

Esta pantalla contiene un componente conectado directamente a un API de geo-localización y visualización de mapas digitales la cual ocupará toda la pantalla, no obstante, se podrá visualizar información adicional de marcadores de ubicación por medio de menús contextuales que proporciona el contexto de aplicaciones en Android.

Adicionalmente se incorpora una barra de herramientas para buscar direcciones o determinar la ubicación actual.

CAPÍTULO V.

PRUEBAS Y RESULTADOS

5.1. INTRODUCCIÓN

En el presente capítulo se evaluarán y analizarán aspectos tanto de rendimiento físico (de Hardware) como la eficacia con la que está trabajando la aplicación en ciertos dispositivos clientes, así como toda la infraestructura donde está implantada la misma, con el objetivo de obtener valores y datos con los cuales se concluya si la aplicación es factible de implementarla en entornos reales y empresariales.

Se tomarán aspecto de redes y telecomunicaciones, acceso a internet, geo posicionamiento, tráfico de datos, velocidad, etc., para las respectivas pruebas tanto en ambientes locales como externos.

Para el proceso se ha montado y configurado un ambiente tanto de desarrollo como para la producción y lanzamiento de la aplicación en un espacio físico el cual sea similar a posibles casos reales donde la aplicación pueda interactuar funcionalmente.

5.2. EQUIPO INFORMÁTICO

Dentro de la implementación de los aplicativos se ha requerido de los siguientes equipos informáticos.

- **Servidor de Base de Datos**
    - Computadora de Escritorio con procesador Intel Core 2 Duo E7500 a 2.3 Ghz. Arquitectura de 64 bits.

- 2 GB de Memoria RAM.
- 160 Gb de Disco Duro.
- Sistema Operatvo: Windows Server 2012
- 1 Tarjeta de red Ethernet.

- **Servidor de Aplicaciones**
  - Computadora de Escritorio con procesador Intel Core i7-4790 a 3.6 Ghz. Arquitectura de 64 bits.
  - 4 GB de Memoria RAM.
  - 1 TB de Disco duro
  - Sistema Operativo Windows Server 2012
  - 1 Tarjeta de red Ethernet

- **Servidor Firewall**
  - Servidor HP ProLiant ML150 G3 con procesador Intel Xeon Dual core 5110 a 1.6 Ghz,
  - 1 MB de Cache, Bus 1333 Mhz
  - 2GB de Memoria RAM
  - 75 GB de Disco Duro RAID
  - Sistema Operativo Linux Debian Untangle
  - 2 Tarjetas de red Ethernet.

- **Dispositivo Cliente**
  - Celular inteligente Samsung Galaxy S4 GT-I9500

- Procesador Exynos 5 Octa octa-core 1.6 Ghz /Qualcomm Snapdragon 600 1.9 Ghz
- 2GB de memoria RAM.
- 32 GB de Disco duro.
- Sistema Operativo Android v4.2.2 Jelly Bean.
- NIC Inalámbrico Wi-Fi, NIC para conexión a datos GSM.

5.3. IMPLEMENTACIÓN

Dentro del proceso de desarrollo y codificación de software se ha requerido varios servicios y librerías de acceso en tiempo real alojadas en la nube, mismas librerías o APIS que han sido de soporte para el servicio de visualización de mapas, mensajería, ubicación etc.

5.3.1. Google Cloud Messaging (GCM API)

"Google Cloud Messaging GCM es un servicio gratuito que permite a los desarrolladores enviar mensajes entre servidores de aplicaciones y aplicativos clientes, esto también incluye los mensajes enviados desde los servidores a los clientes y viceversa". (Orero, 2016)

GCM tiene la funcionalidad de utilizar los protocolos XMPP (Protocolo extensible de mensajería y comunicación de presencia) y HTTP (Protocolo de transferencia de Hiper texto) sea para envió o recepción de mensajes respectivamente y su funcionamiento está reflejado en la Ilustración V-1.

Ilustración V-1: Arquitectura GCM

Elaborado por: Diego Ponce – Juan Prado

Basado de: Google Inc.

Los protocolos antes mencionados colaboran para que la tecnología "Push" sea posible, la cual un servidor tiene la capacidad de enviar mensajes a un receptor de mensajes, cliente, consumidor, etc. Sea el caso siempre el receptor debe estar registrado al grupo de notificaciones "Push" donde opera el servidor.

El desarrollador puede hacer uso gratuito de la plataforma GCM para la gestión de envío/recepción de mensajes entre clientes y servidores, la plataforma antes mencionada estará encargada de administrar la cola de mensajes tanto enviados como recibidos y re direccionarlos a su respectivo destino.

El escenario de funcionamiento de envío y recepción de mensajes por medio de GCM deben intervenir por lo menos 3 componentes principales, El servidor GCM en la nube, un dispositivo móvil con su aplicación de mensajes "Push" respectiva, y un servidor receptor de mensajes, la cual gestionará o guardará esta información según sea necesario.

Dentro de los paquetes de mensajes que gestiona y re direcciona GCM existen 4 credenciales importantes que deben ser tomadas en cuenta, ya que, a base de estas, los mensajes serán entregados al destinatario correcto. (Orero, 2016)

- **“Sender ID”:** La aplicación móvil que contiene tecnología “Push” debe ser registrada en la plataforma GCM, este ID servirá para reconocer dicha aplicación.
- **“Application ID”:** Identificador con el cual la aplicación móvil está lista para recibir mensajes, este identificador es configurado en el archivo “.manifest” en la arquitectura de proyecto de desarrollo de Android.
- **“Registration ID”:** Este ID es enviado por la plataforma GCM desde la nube, después de que la aplicación móvil haya hecho una solicitud de registro a éste, no obstante, debe ser registrado en un servidor tercero, ya sea corporativo o de pruebas para tener el respectivo registro del ID correspondiente a una aplicación determinada vinculado a dispositivo determinado.
- **“Sender Auth Token”:** Identificador con el cual el servidor tercero podrá tener acceso a los servicios de Google y sus API’s como tales.

Con todas las credenciales necesarios para la correcta trasmisión de mensajes “Push” por GCM se puede definir su funcionamiento el cual se indica en la Ilustración V-2.

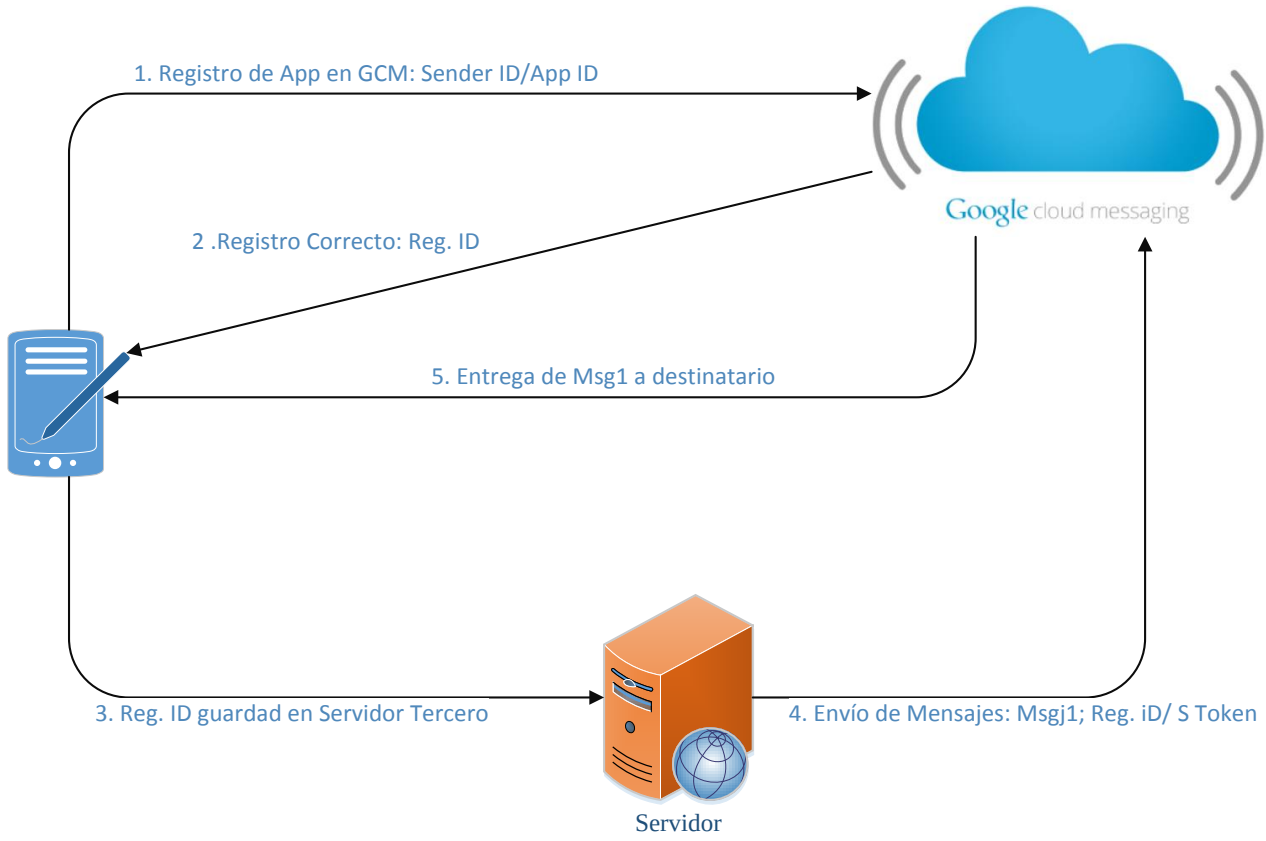


Ilustración V-2: Arquitectura de funcionamiento de mensajes "Push" mediante GCM

Elaborado por: Diego Ponce – Juan Prado

Basado de: (Orero, 2016)

5.1.1. Google Maps (GMAPS API)

Dentro del mundo de desarrollo de software existen varias compañías grandes e importantes que se dedican a dar diversos servicios, soporte y herramientas (API's) como apoyo de la creación de aplicaciones de escritorio, web, móviles, etc., entre estos recursos se encuentran herramientas o componentes que son utilizados para aplicaciones o funcionalidades SIG (Sistemas de Información Geográfica), de las cuales existen varias opciones como Virtual Earth de Microsoft, Yahoo! Maps de Yahoo!, GMaps de Google entre

otros. (Antoni Pérez Navarro, Albert Botella, Anna Muñoz Bollas, Rosa Olvella González, Joan Carlos Olmedillas Hernandez, Jesús Rodrigues Lloret, 2011)

En el presente apartado se define el uso de Google Maps como API de desarrollo tanto para la aplicación móvil como para la plataforma web tomando en cuenta las siguientes razones de uso.

- Total compatibilidad con desarrollo de aplicaciones Android y JSF como tal.
- Google Maps como herramienta SIG con mayor demanda a nivel mundial.
- Funcionalidades diversas para edición y configuración de datos geográficos.
- Factibilidad de integración en cualquier plataforma de desarrollo.
- Soporte y visión de evolución de la plataforma a largo plazo.

El API GMaps básicamente permite visualizar mapas digitales en tiempo real desde los servidores de Google en nuestras aplicaciones distribuidas, móviles o páginas web mediante un código determinado, ya sea java script, java, etc.

La política de uso y privacidad del API es libre y gratuita, no obstante, está puede variar si el sitio o la aplicación sobrepasa un cierto número de solicitudes de uso o visualización (500000 solicitudes aprox.), sea el caso la página web o aplicación tendrá notificaciones para no perder la calidad de la misma en sus servidores.

Como dato adicional, se debe solicitar un registro y autorización a Google en forma de un identificador de aplicación (API key) para el acceso y uso de Google Maps en una aplicación o sitio web predeterminado.

El funcionamiento y configuración de API se explica en sencillos pasos, ya que su programación puede variar dependiendo el entorno de desarrollo y lenguaje que se utilice.

- Definición de la interfaz, página o módulo donde se quiere implementar la visualización del mapa digital en la aplicación o página web sea el caso.
- Dentro del código de programación declarar las librerías necesarias para el funcionamiento de Google Maps, en muchos de los casos se requiere un identificador de aplicación (API key), generada con el registro del proyecto en las páginas de “Google developer” para tener acceso a las funcionalidades de mapas digitales.
- Generación del código para la creación y configuración del mapa digital.
- Definir funcionalidades adicionales para la interfaz del mapa como ubicación actual, opciones de escala, buscador de direcciones y lugares, etc.

5.4. CODIFICACIÓN

5.4.1. Estándares de Programación

5.4.1.1. Introducción

El presente proyecto de desarrollo de software el cual es de apoyo para el estudio de las redes Wi-Fi y GPS ha implementado en el lenguaje de programación JAVA 2EE y tomando en cuenta que la aplicación tiene factibilidad de implantarse a nivel corporativo o comercial sea el caso se requiere de estándares y convenios de programación justificando los siguientes motivos.

- El costo de la aplicación y su código puede influir en el mantenimiento y actualizaciones requeridas, un promedio del 75% de los beneficios económicos.

- Un estándar o convenio de programación servirá para el buen entendimiento y efectividad del desarrollo de software.

- El código puede tener la factibilidad de patentarse y distribuirse para lo cual necesitará cumplir con estándares de programación.

- Eventualmente, el proyecto puede re asignarse a otros programadores, los cuales necesitarán la debida documentación y estandarización para continuar el proyecto.

5.4.1.2. Sobre los Nombres de los Archivos

5.4.1.2.1. Extensiones

Se utiliza las extensiones predeterminadas por Java

- Extensión para archivos codificados en Java: *.java
- Extensión para archivos compilados de Java: *.class

5.4.1.2.2. Nombres comunes de archivos

Dentro de los archivos más utilizados en esta sección se encuentran:

- **"GNUmakefile":** Se usa de manera común archivos con el nombre "make". Se usa "gnumake" para construir el software.

- **README:** Nombre comúnmente utilizado para crear referencia del contenido de un paquete o fichero predeterminado.

5.4.1.3. Sobre la organización de los Ficheros

Al momento de crear archivos y organizarlos se toman en cuenta en sus contenidos deben ser clasificados por líneas en blanco y con los respectivos comentarios para identificar la función de cada sección del código en el archivo.

El contenido de los archivos será limitado a máximo 2000 líneas de código justificando su facilidad y comodidad para manejarlos.

5.4.1.3.1. Sobre los archivos de código en Java

Todos los archivos de código fuente de Java almacenarán un solo elemento de identificación ya sea una única interfaz del archivo o clase pública sea el caso.

En el caso de existir varias clases o interfaces que deben relacionarse, se los puede organizar en el mismo archivo de la clase pública.

Adicional, los archivos de código fuente deben contener el siguiente contenido y orden.

- Documentación o comentarios de introducción o inicio del fichero.
- Sentencias de código de métodos de declaración de paquetes e importe de librerías o clases: “Package” e “Import”.
- Las respectivas declaraciones de interfaces y clases.

- **Comentarios de Introducción**

En todos los archivos de código fuente en Java comenzará con un comentario en el que se identificará información básica de la clase como el nombre, versión, fecha y licencia de la misma.

```
/*
*Nombre de la clase o archivo
*
*Número o información de la versión
*
*Fecha de Creación y modificación del archivo o clase
*
*Información de Licencia o Copyright del archivo o clase
*/
```

Ilustración V-3: Comentarios de Introducción

Elaborado por: Diego Ponce – Juan Prado

- **Sentencias de Paquete e Importación de librerías o clases**

Después del comentario de introducción se define el paquete al que pertenece la clase o archivo con la sentencia "Package", a continuación, puede implementarse varias sentencias "Import" para usar las librerías requeridas por la clase.

```
package java.awt.main_package;
import java.awt.library_folder.*;
import java.awt.library_follder2.*;
```

Ilustración V-4: Sentencias de Paquete e Importación de Librerías o clases

Elaborado por: Diego Ponce – Juan Prado

5.4.1.3.2. Sobre las declaraciones de Interfaces y clases

A continuación, se muestra la forma de declaración de interfaces o clases para archivos de código fuente, y el orden con el cual debe usarse.

Tabla V-1: Declaraciones de Interfaces y Clases

No.	Descripción de Declaración	Observaciones
1	Comentario corto para documentación de interfaz o clase (/**…*/).	
2	Sentencias de código "interface" ó "class" .	
3	Comentario para la documentación de interfaz o clase sea el caso de requerirlo.	Este tipo de comentarios debe ser general si necesidad de especificar la funcionalidad de las demás.
4	Variable de tipo estático (static)	El orden de uso de estas variables es la declaración inicial de tipo públicas (public), después las protegidas (protected) y finalmente las privadas (private)
5	Variables de tipo Instancia	Su orden de uso es similar a las anteriormente mencionadas 1. public, 2. protected, 3. private
6	Sentencias constructores	
7	Sobre los métodos	Su implementación debe clasificarse por la funcionalidad al que se le otorga, con la finalidad de más legibilidad y comprensibilidad en el

		código

Elaborado por: Diego Ponce – Juan Prado

5.4.1.4. Sobre la Indentación o Sangrado

En cuanto al sangrado usualmente se usarán 4 espacios de línea como referencia, de igual manera en las tabulaciones se tendrá como referencia los cuatro espacios.

5.4.1.4.1. Longitud de Línea dentro del código fuente

Cada línea de código debe tener como máximo 80 caracteres por el motivo que mucha de ellas podría ya no reconocer en ciertos entornos de ejecución o IDE's de desarrollo.

5.4.1.4.2. Ruptura de Línea de código

En muchas ocasiones o estructuras de código pueden ser interrumpidas por un salto de línea, en aquel caso se puede lo puede aplicar considerando los siguientes estipulados.

- Solo se puede romper una línea después de una coma.

- Es recomendable alinear expresiones de código al mismo nivel de la línea anterior del mismo.

- Se debe tomar en cuanta romper la línea de código solamente antes de un operador.

- Se preferible considerar las rupturas de alto nivel que las de bajo nivel, más a la derecha que la izquierda que el padre.

- En el supuesto caso que los estipulados anteriormente mencionados influyeran para la ilegibilidad del código fuente, se tomará una Indentación de 8 espacios por tabulación.

```
metodoMuestra(exp1, exp2, exp3,
              exp4, exp5, exp6);

var= metodoMuestra1(exp1,
                    metodoMuestra3(exp2,
                    exp3));
```

Ilustración V-5: Rupturas de Línea de Código

Elaborado por: Diego Ponce – Juan Prado

También se puede considerar romper líneas de código en expresiones lógicas y aritméticas se prefiere realizar un salto de línea que ocurre fuera de la expresión que encierra los paréntesis.

```
var1 = var2 * (var3 + var4 + var5)
     + 5 * var6; //PREFERIR ESTE MODO

var1 = var2 * (var3 + var4
     - var1) +4 * var6; //EVITAR ESTE MODO
```

Ilustración V-6: Ruptura de Línea de Código – Expresiones Lógicas y Aritméticas

Elaborado por: Diego Ponce – Juan Prado

La ruptura de líneas de código para estructuras condicionales IF tendrá la característica de seguir los 8 espacios de tabulación, con 4 espacios es difícil ver la estructura de manera legible.

```
//NO USAR ESTA INDENTACIÓN
if ((condicion1 && condicion2)
    || (condicion3 && condicion4)
    || (condicion5 && condicion6){
    acciones();
    }
//O ESTA INDENTACIÓN
if((condicion1 && condicion2) || (condicion3 && condcion4)
        || condicion5 && condicion6) {
    acciones();
        }
```

Ilustración V-7: Ruptura de Código – Estructura de Datos

Elaborado por: Diego Ponce – Juan Prado

Formas de expresiones ternarias para desarrollo de código fuente

```
var = (expresionlargaBol) ? var2 : var3;

var = (expresionlargaBol) ? var2
                          : var3;

var = (expresionlargaBol)
      ? var2
      : var3;
```

Ilustración V-8: Formas de expresiones ternarias para el desarrollo de código fuente

Elaborado por: Diego Ponce – Juan Prado

5.4.1.5. Sobre los comentarios

Dentro de la codificación del proyecto se han definido dos clases de comentarios para la documentación del mismo

- Comentarios de implementación, delimitados por /*…*/.
- Los comentarios de documentación, son delimitados por /**…*/, de igual manera pueden ser exportados a formado HTML por medio de la herramienta javadoc, suelen servir para desarrolladores que no tienen el código disponible.

Los comentarios deben siempre estar delimitados en base a los principios antes mencionados y sin decoraciones o demasiados asteriscos.

5.4.1.5.1. Formato para los comentarios de implementación en el código fuente

Se ha define cuatro formatos o estilos para los comentarios de implementación en los cuales destacan: de bloque, de una línea, finales, y de fin de carro o línea.

- **Comentarios de Bloque para comentarios de Implementación**

Estos estilos de comentarios son usados mayormente para describir los archivos de código, funciones o métodos, estructuras de datos o condicionales, y ciertos algoritmos que pueden tener un grado de comprensión complejo.

No obstante, también pueden ser usados dentro de funciones o métodos, para definir alguna línea de código destacable o necesaria para otros desarrolladores.

Adicionalmente, se recomienda que los comentarios de bloque estén precedidos por un salto de línea completamente en blanco que pueda separarlo moderadamente del código.

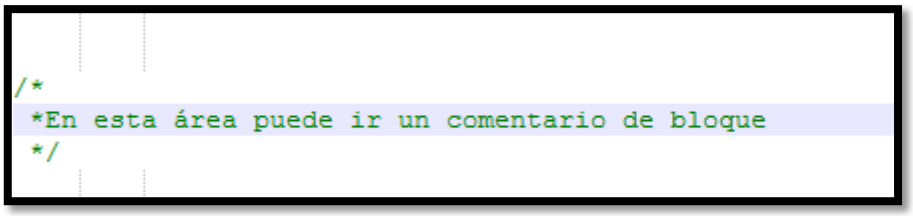


Ilustración V-9: Comentario de Bloque

Elaborado por: Diego Ponce – Juan Prado

Un comentario de bloque puede identificarse con el modificador /*- el cual indica que el bloque no puede ser formateado, también se lo conoce como “indent(1)”.

```
/*-
*
*En esta comentario de bloque se ha definido
*un formato especial predeterminado que puede ignorarse indent(1)
*
*  uno
*      dos
*          tres
*
*/
```

Ilustración V-10: Comentario de Bloque Indentado

Elaborado por: Diego Ponce – Juan Prado

- **Comentarios de Líneas Simples**

Puede darse el caso de requerir comentarios cortos que puntualicen descripciones, notas u observaciones en una línea determinada del código, el cual debe ir precedido de una línea en blanco.

```
if (condicion1){
    /*nota o descripción de la condición */
    ...
}
```

Ilustración V-11: Comentario de Líneas Simples

Elaborado por: Diego Ponce – Juan Prado

- **Comentarios Finales**

El identificador de escape "//" tiene la capacidad de convertir en comentario a una línea dentro del archivo de código fuente o bien sea parte de ella, no obstante, su uso está excluido para comentarios de varias líneas consecutivas a no ser que fuera para comentar varias líneas de código.

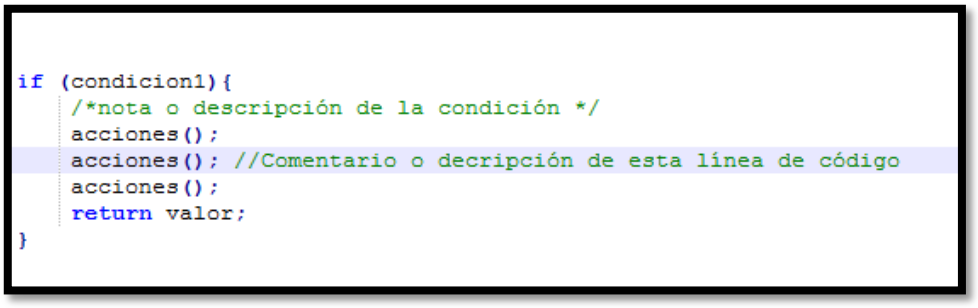


Ilustración V-12: Comentario de una sola línea

Elaborado por: Diego Ponce – Juan Prado

5.4.1.5.2. Comentarios de Documentación

Este tipo de comentarios hace referencia a la descripción o contenido de las clases, constructores, interfaces, funciones o métodos en él. Están delimitados por /**…*/, el cual debe estar justamente donde se va a hacer la declaración.

```
/*
 *La siguiente clase tiene como funcionalidad...
 */
public class Clase1{...
...
}
```

Ilustración V-13: Comentario de Documentación

Elaborado por: Diego Ponce – Juan Prado

Si los comentarios tienen como finalidad ser usados como referencia para toda la documentación, estos no deben ir dentro de un método o después o en el interior de un constructor, ya que javadoc puede confundirse al clasificar entre el comentario de la documentación y cualquier otro.

5.4.1.6. Declaraciones

5.4.1.6.1. Cantidad por línea de código

Se sugiere las declaraciones que sea una por línea, pues esto facilitará la documentación y los comentarios dentro del código.

```
int var1; //Descripción de la variable 1
int var2; //Descripción de la variable 2

//y evitar...

int var1, var2;
```

Ilustración V-14: Declaraciones de Cantidad por Línea de Código

Elaborado por: Diego Ponce – Juan Prado

Tampoco se recomienda hacer declaraciones del mismo tipo dentro de la misma línea de código.

5.4.1.6.2. Inicialización dentro de las declaraciones

Es recomendable inicializar absolutamente todas las variables locales declaradas en un archivo código, a no ser que, dependiendo de otras variables con cálculos posteriores, esta primera se vea afectada.

5.2.1.4.5. Colocación de las declaraciones

Las declaraciones deben hacerse sólo al principio de bloques de estructuras de datos o justamente después de empezar a codificar una clase, es decir luego de la primera llave.

```
void miMetodo(){
    int entero1 = 0; //Inicio del bloque del método
    if(condicion){
        int entero 2 = 0; //comienzo del bloque del "if"
        ...
    }
}
```

Ilustración V-15: Colocación de declaraciones

Elaborado por: Diego Ponce – Juan Prado

Sin embargo, puede generase una excepción para las estructuras “for”, pues en el lenguaje de java pueden declararse dentro de la misma estructura.

```
for(int i = 0; i < ContadorMaximo(); i++){
    ...
}
```

Ilustración V-16: Declaraciones Lazos "For"

Elaborado por: Diego Ponce – Juan Prado

Es recomendable evitar declaraciones dentro de estructuras de datos, es decir de niveles superiores e innecesarios.

```
void miMetodo(){

    if(condicion){
        int entero 2 = 0; //EVITAR ESTE MODO
        ...
    }
}
```

Ilustración V-17: Excepciones de declaraciones en estructura de Datos

Elaborado por: Diego Ponce – Juan Prado

5.2.1.4.6. Declaraciones con respecto a las Interfaces o Clases

Se debe tomar en cuenta los siguientes principios al programar declaraciones para clases o interfaces.

- No se debe agregar espacios en blanco después del nombre del método implementado y antes de declarar el paréntesis, el cual inicia los parámetros predeterminados requeridos.

- La llave de apertura del método "{"se implementaría al final de la declaración anteriormente hecha.

- De igual manera, la llave de cierre para bloques o métodos"}" está en indentada justamente después de la última línea del método.

- Cada método suele separarse con un salto de línea en blanco.

```
class Clase1 extendes Object {

    int variable1;
    int variable2;

    Clase1(int i, int j){
        var1 = i;
        var2 = j;
    }

    int metodo1() {}
    ...
}
```

Ilustración V-18: Declaraciones en Interfaces o Clases

Elaborado por: Diego Ponce – Juan Prado

5.4.1.7. Sentencias

5.4.1.7.1. Sentencias Simples

Se define que cada línea de programación debe tener como mucho una sola línea de la misma.

```
var1++;                    //Correcto

var2+=var1;                //Correcto

var3++; var2=+var3;        //Incorrecto
```

Ilustración V-19: Uso de sentencias simples

Elaborado por: Diego Ponce – Juan Prado

5.4.1.7.2. Sentencias Compuestas

Este tipo de sentencias se caracterizan por contener otras sentencias dentro de llaves "{acciones}", y deben ser usadas de acuerdo a los siguientes estipulados.

- Las sentencias que están encerradas en otra deben tener un sangrado en un nivel superior a la sentencia que la encierra.

- La llave de introducción de la sentencia compuesta debe estar al final de esta, en cambio, la llave de cierre debe permanecer al final de las sentencias dentro de la estructura compuesta, y al mismo nivel de donde empezó esta.

- A pesar de que pueda ser una sentencia simple, es recomendable siempre usar llaves, con el fin de evitar errores de código tanto al momento de compilar como de ejecutar el programa.

5.4.1.7.3. Sentencias de retorno

Los valores de retorno siempre deben ir entre paréntesis, a no ser que excepcionalmente devuelva un valor obvio para el método.

```
return;

return Metodo.tam();

return (tam ? tam : tamdef);
```

Ilustración V-20: Uso de sentencias de retorno

Elaborado por: Diego Ponce – Juan Prado

5.4.1.7.4. Sentencias condicionales if, if-else, if else-if else

Este tipo de sentencias condicionales deben tener el siguiente formato.

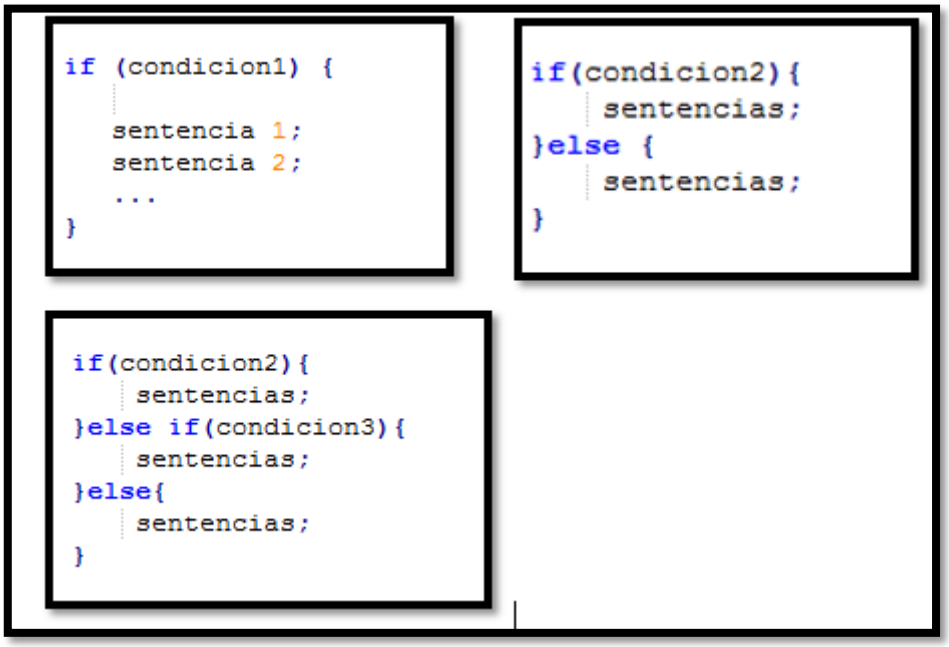


Ilustración V-21: Uso de sentencias condicionales

Elaborado por: Diego Ponce – Juan Prado

5.4.1.7.5. Sentencias de bucle for

Esta sentencia de lazo debe implementarse de la siguiente manera.

```
for(inicializacion; condicion; lazo){
    sentencias;
}
```

Ilustración V-22: Uso de sentencias Bucle "For"

Elaborado por: Diego Ponce – Juan Prado

Un bucle for sin contenido puede implementarse de la siguiente forma.

```
for(inicializacion; condicion; lazo);
```

Ilustración V-23: Uso de "For" en sentencias simples

Elaborado por: Diego Ponce – Juan Prado

5.4.1.7.6. Sentencias de bucle while

Para este tipo de lazos es recomendable utilizarlo de la siguiente forma.

```
while (condicion1){
    sentencias;
}

while; //Sentencias While vacía
```

Ilustración V-24: Uso de lazo "While"

Elaborado por: Diego Ponce – Juan Prado

5.4.1.7.7. Sentencias de bucle Do-While

En este tipo de bucles se recomienda basarse en el siguiente ejemplo.

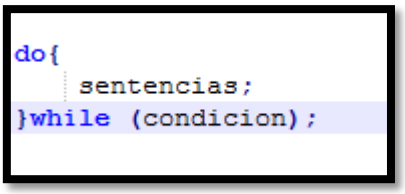


Ilustración V-25: Uso de Lazos "Do-WHile"

Elaborado por: Diego Ponce – Juan Prado

5.4.1.7.8. Sentencias de estructura de datos Switch

Esta estructura de datos debe basarse en el siguiente esquema.

```
switch (condicion){

Case 1:
     sentencias;
     /* Caso de propagación */
Case 2:
     sentencias;
     break;
Case 3:
     sentencias;
     break;
Default:
     sentencias;
     break;
}
```

Ilustración V-26: Uso de estructuras "Switch"

Elaborado por: Diego Ponce – Juan Prado

Observe que en el caso de propagación es necesario siempre identificarlo o aclararlo con un comentario, además, cada sentencia o grupo de ellas debe terminar con un "break;"

5.4.1.7.9. Sentencias de gestión de excepciones Try-Catch

Este tipo de sentencias debe usarse de la siguiente manera.

```
try{
    sentencias;
}catch(Excepcion e){
    sentencias;
}
```

Ilustración V-27: Uso de Captura de Excepciones "Try-Catch"

Elaborado por: Diego Ponce – Juan Prado

Estas sentencias pueden implementarse junto con el comando “finally”, aunque su ejecución seguirá siendo independiente de esta.

```
try{
    sentencias;
}catch(Excepcion e){
    sentencias;
}finally{
    sentencias;
}
```

Ilustración V-28: Uso de sentencia “Finally”

Elaborado por: Diego Ponce – Juan Prado

5.4.1.8. Espacios en Blanco

5.4.1.8.1. Líneas en blanco dentro del código fuente

Este hábito de programación puede llegar a mejorar la comprensión del código fuente por usuarios ajenos a los autores, para utilizarlo se debe tomar en cuenta los siguientes puntos.

- Usar entre clases o interfaces.
- Para clasificar secciones de un archivo de código fuente.
- Para clasificación de métodos.
- Antes de un comentario de sección de bloque o de línea simple.

- Para la clasificación de las secciones según el desarrollador.

5.4.1.8.2. Espacios en Blanco

Las condiciones para el uso de espacios en blanco son las siguientes.

- Palabras reservadas del lenguaje y a continuación el paréntesis debe emplear espacio en blanco

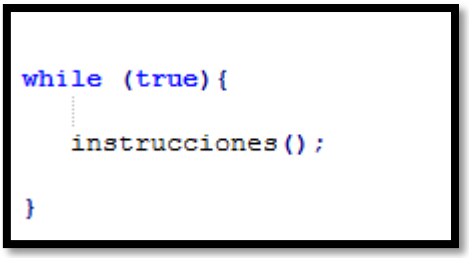


Ilustración V-29: Uso de espacios en blanco

Elaborado por: Diego Ponce – Juan Prado

- Debe emplearse los espacios en blanco a continuación, después del uso de una coma ya sea para requerimiento o envió de listas de parámetros de un método.

- Todos los operadores a diferencia del operador lógico binario ".", tampoco debe utilizarse espacios para separar operadores que estén situados consecutivamente.

- Los lazos for deben separarse con espacios en blanco de la siguiente manera.

```
for (exp1; exp2; exp3)
```

Ilustración V-30: Uso de espacios en blanco en lazos "For"

Elaborado por: Diego Ponce – Juan Prado

- De igual manera las sentencias "Cast" deben emplearse con espacios en blanco.

```
metodo1((objetc) unNumero, (Objetc) x);
metodo2((vartype) (cp + 4), ((int) (i + 6)) +3);
```

Ilustración V-31: Uso de espacios en blanco en Casitngs

Elaborado por: Diego Ponce – Juan Prado

5.4.1.9. Estándares sobre la nomenclatura

Tipo de Identificador	Reglas de Nomenclatura	Ejemplos

Este tipo de convenciones define si el código fuente desarrollado es legible o no para otros desarrolladores, adicionalmente, define referencias de identificadores, variables, constantes, declaraciones, etc.

Package (Paquetes)	Para el nombre de este identificador solo se debe emplear el set de caracteres ASCII en minúscula, el contexto de su uso debe ser de alto nivel, tal cual se especifica en el estándar OSI 3166, 1981. Los nombres de los paquetes deben referir a una idea clara del contenido de las mismas, o su dominio sea el caso.	`com.sun.eng`
		`com.apple.media.quick.v5`
		`edu.gov.ec.product`

Clases	Con respecto a las clases siempre deben usarse sustantivos, en el caso de ser nombres compuestos se deberá empezar la primera letra de la palabra complementaria con letra capital, tenga en cuenta nombres sencillos y de acuerdo al contexto de su contenido, evitar palabras de tipo acrónimo o abreviaturas	`class Alumno;` `class RutaPaquete;`
Interfaces	La nomenclatura de interfaces es igual a la de las clases anteriormente mencionadas	`interface VariableObjeto;` `Interface Alumno;`

Métodos	Este tipo de sentencias se deben implementarlas con verbos. En caso de ser palabras compuestas, el termino complementario debe implementarse con letra capital.	`ejecucion();`
		`metodoEjecucion();`
		`lanzarInstancia();`

Variables	A excepción de las constantes, declaraciones y variables se utilizarán términos en minúscula, en caso de ser compuestas, el termino complementario se usará letra capital, no deben empezar con guion bajo (_) o signo de dólar ($). Deben ser cortos y puntuales en su contexto de uso. Finalmente, solo debe usarse variables de un solo caracter, para uso temporal o de indexación.	`int a;`
		`char c;`
		`boolean Bandera;`
		`boolean miBandera;`

Constantes	Estas en cambio a las anteriores deben ir en mayúsculas y separando términos compuestos con guión bajo (_).	`static final float ALTURA_MINIMA = 6;`
		`static final float ALTURA_MAXIMA = 599;`
		`static final boolean ES_VERDADERO = true;`

Tabla V-2: Estándares sobre la nomenclatura

Elaborado por: Diego Ponce – Juan Prado

5.4.1.10. Bueno hábitos de la programación

5.4.1.10.1. Gestión del acceso de variables, e instancias de la clase

El hecho de liberar el acceso a una variable y hacerla pública debe ser claramente justificada con alguna buena razón, los procesos para hacerlo siempre deben ser con la implementación de métodos "get" y "set", no obstante, no todas las variables necesitarán ser accedidas.

5.4.1.10.2. Hábitos de referencia a variables y métodos de la clase

Se evitará el uso de objetos para tener acceso a variables o métodos como son la de tipo estática (static). Es recomendable usar el nombre de la clase.

```
memtodoDeClase();                    //Correcto
ClasePrincipal.MetodoDeClase();      //Correcto
objeto.metodoDeClase;                //Incorrecto
```

Ilustración V-32: Hábitos de referencia a variables y métodos de la clase

Elaborado por: Diego Ponce – Juan Prado

5.4.1.10.3. Hábitos con respecto a las constantes

Si bien pueden implementarse constantes de todo tipo, es recomendable hacerlos para valores específicos como pueden ser enteros binarios como 1, 0, -1 que pueden utilizarse como bandera en sentencias de bucle o lazo.

5.4.1.10.4. Hábitos con respecto a la asignación de variables

No es buen hábito reasignar valores a variables y nuevamente a otras en la misma sentencia de código.

```
Objeto.variable = obejto.variable2 = '2'; //Incorrecto
```

Ilustración V-33: Hábitos con respecto a la asignación de variables

Elaborado por: Diego Ponce – Juan Prado

No usar el operador "=" como asignación en una expresión, ya que puede confundir a otros desarrolladores abduciendo que es una comparación lógica.

```
if(a++ = i++){ //Incorrecto
    instrucciones;
}
```

Ilustración V-34: Hábitos no adecuado en el uso de operadores

Elaborado por: Diego Ponce – Juan Prado

Su manera de uso debería ser.

```
if((a++ = i++)!=0){     //Corrrecto
    instrucciones;
}
```

Ilustración V-35: Hábitos adecuados en el uso de operadores

Elaborado por: Diego Ponce – Juan Prado

Tampoco es recomendable usar asignaciones dentro de otras en forma embebida

```
res = (a = c) + sum;    //Incorrecto
res = a +b;             //Correcto
a = a + sum;            //Correcto
```

Ilustración V-36: Hábitos en el uso de formas embebidas de programación

Elaborado por: Diego Ponce – Juan Prado

5.4.1.10.5. Hábitos adicionales

- **Con respecto al uso de paréntesis**

Es recomendable la implementación de paréntesis en expresiones que tengan un grado de complejidad alto al momento de comprenderlos y que pueda ser legible para otros desarrolladores.

```
if (a == b || b == c)       //Incorrecto
if ((a == b) ||( b == c))   //Correcto
```

Ilustración V-37: Hábitos con respecto al uso de paréntesis

Elaborado por: Diego Ponce – Juan Prado

- **Con respecto a los valores de retorno**

Es recomendable que la estructura de sus métodos o su código en general haga referencia al objetivo que sigue la sentencia, y no darle funciones innecesarias;

```
if(expLogica()){                    //Correcto
    return verdadero;
}else{
    return falso;
}

return expLogica;                   //Incorrecto

if(condicion1){                     //Correcto
    return var1;
}

return varAux;

return (condicion1 ? x : y); //Incorrecto
```

Ilustración V-38: Hábitos con respecto al uso de valores de retorno

Elaborado por: Diego Ponce – Juan Prado

- **Con respecto a expresiones antes del signo "?" como operador condicional**

Debe realizarse el uso de paréntesis para las condiciones antes del signo ternario "?" si el caso es usarlo como sentencia condicional.

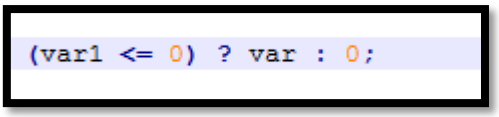


Ilustración V-39: Hábitos con respecto al uso de expresiones condicionales con operadores

Elaborado por: Diego Ponce – Juan Prado

- **Con respecto a los comentarios especiales**

Dentro de la documentación del código fuente se recomienda usar <<XXX>> en validar el comentario con algún error en específico, en cambio para indicar el error tanto de compilación y ejecución en alguna sentencia o grupo de ella, se utilizará <<FIXME>>

5.4.2. Desarrollo de Código Fuente

5.4.2.1. Codificación Monitorización Virtual Administración Web

5.4.2.1.1. Estructura de la aplicación Monitorizacion Virtual Administración Web

La estructura de la monitorización virtual Adminitración Web se la desarrollo bajo los siguientes parámetros la cual se muestra en la Figura V-40.

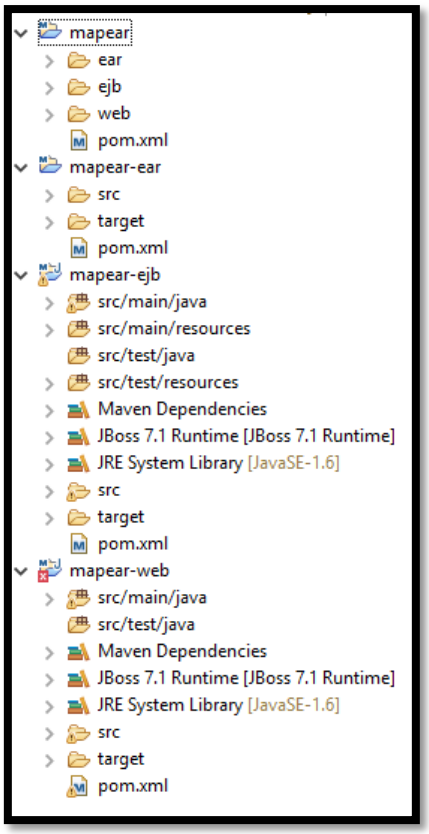


Ilustración V-40: Estructura de la aplicación de Monitorización Virtual de Administración Web

Elaborado por: Diego Ponce – Juan Prado

- **Mapear-ear y Mapear**: Estos directorios organizarán de manera coherente los módulos que serán ejecutados por el servidor de aplicaciones. Tiene archivos XML que son nombrados como descriptores de despliegue. En este proyecto en específico guardará tanto los archivos .war y .jar para su correcto funcionamiento en conjunto.

- **Mapear-ejb:** Este directorio al momento de ser empaquetado por el compilador lo empaqueta en formato .jar. El mismo especifica que en ese directorio se va a trabajar con EJBs de diferentes tipos las cuales son:

- **Entidades:** Estas entidades están mapeadas a través de "hibernate" las cuales hacen conexión directa con la base de datos y el modelo de objetos que se va a utilizar en la aplicación.

- **DAO:** Permiten hacer transacciones con las entidades a través de un DAO genérico y HQL.

- **Mapear-web:** Este directorio nos permitirá empaquetar en formato .war todas las vistas que percibirá el cliente con su respectivo Bean para la conexión entre el modelo de base de datos con su respectivo DAO.

5.4.2.2. Configuración de servidor de aplicaciones Jboss

Para la implementación de la aplicación Monitorización Virtual Administración Web es necesario configurar el archivo "standalone.xml".

5.4.2.2.1. Archivo de configuración "Standalone.xml"

El archivo se encuentra ubicado en /standalone/configuración/ y su nombre es standalone.xml.

Es necesario configurar dentro de este archivo la conexión de la base de datos como se muestra en la siguiente figura.

```
<datasource jta="true" jndi-name="java:jboss/mapping-ds" pool-name="mappingApp" enabled="true" use-ccm="false">
    <connection-url>jdbc:mysql://localhost:3306/pucemappdb</connection-url>
    <driver-class>com.mysql.jdbc.Driver</driver-class>
    <driver>mysql</driver>
    <security>
        <user-name>root</user-name>
        <password>root</password>
    </security>
    <validation>
        <validate-on-match>false</validate-on-match>
        <background-validation>false</background-validation>
    </validation>
    <statement>
        <share-prepared-statements>false</share-prepared-statements>
    </statement>
</datasource>
<drivers>

    <driver name="mysql" module="com.mysql">
        <xa-datasource-class>com.mysql.jdbc.jdbc2.optional.MysqlXADataSource</xa-datasource-class>
    </driver>
</drivers>
```

Ilustración V-41: Archivo de Configuración "Standalone.xml"

Elaborado por: Diego Ponce – Juan Prado

Es importante considerar el nombre JNDI ya que este es el nombre por el cual hibernate hará el mapeo de entidades dentro del proyecto.

5.4.2.2.2. Archivo "Persistence.xml"

El archivo persistence.xml nos ayudará a mapear con hibernate todas las entidades presentes dentro de la base de datos por lo que es necesario especificar el nombre JNDI presente en la configuración del servidor.

La figura siguiente muestra la configuración presente dentro de la aplicación.

```xml
<persistence version="2.0"
   xmlns="http://java.sun.com/xml/ns/persistence" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="
        http://java.sun.com/xml/ns/persistence
        http://java.sun.com/xml/ns/persistence/persistence_2_0.xsd">
   <persistence-unit name="primary">
      <!-- If you are running in a production environment, add a managed
         data source, this example data source is just for development and testing! -->
      <!-- The datasource is deployed as <EAR>/META-INF/mapear-ds.xml, you
         can find it in the source at ear/src/main/application/META-INF/mapear-ds.xml -->
      <jta-data-source>java:jboss/mapping-ds</jta-data-source>
      <class>com.map.entities.Enterprise</class>
      <class>com.map.entities.Form</class>
      <class>com.map.entities.Geolocalization</class>
      <class>com.map.entities.Group</class>
      <class>com.map.entities.Message</class>
      <class>com.map.entities.Role</class>
      <class>com.map.entities.User</class>
      <class>com.map.entities.UserRole</class>
      <properties>
         <!-- Properties for Hibernate -->
         <!--<property name="hibernate.hbm2ddl.auto" value="create-drop" />-->
         <property name="hibernate.show_sql" value="false" />
      </properties>
   </persistence-unit>
</persistence>
```

Ilustración V-42: Archivo "Persistence.xml"

Elaborado por: Diego Ponce – Juan Prado

5.4.2.3. Entidades

Como pudimos observar en la configuración del archivo persistence, este xml específica que clase está relacionado con cierta tabla de la base de datos. Para ello también es necesario configurar la clase de la entidad. Como ejemplo utilizaremos la clase Message que se muestra en la siguiente figura para observar como es la configuración de la misma.

```java
package com.map.entities;

import java.io.Serializable;


/**
 * The persistent class for the message database table.
 *
 */
@Entity
@Table(name="message")
public class Message implements Serializable {
    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    @Column(name="MES_ID")
    private int mesId;

    @Temporal(TemporalType.TIMESTAMP)
    @Column(name="MES_RECEIVED_DATE")
    private Date mesReceivedDate;

    @Temporal(TemporalType.TIMESTAMP)
    @Column(name="MES_SEND_DATE")
    private Date mesSendDate;

    @Column(name="MES_STATE")
    private String mesState;

    @Column(name="MES_TEXT")
    private String mesText;
```

Ilustración V-43: Entidades de programación de la plataforma Web

Elaborado por: Diego Ponce – Juan Prado

Se debe tomar en cuenta los siguientes puntos:

- Es necesario especificar que la clase es una entidad con @Entity
- @Table muestra que tabla de la de datos está relacionada la clase en este caso a la tabla con nombre “message”.
- Si la tabla tiene un atributo con primary key se la especifica con @Id, además si la misma es auto incrementable se la especifica con @GeneratedValue(strategy=GenerationType=IDENTITY)
- Dentro de la clase también hay que declarar atributos y el nombre de la columna de la base datos con la cual hace relación atraves de la sentencia @Column

5.4.2.3.1. Ejb

La declaración de los ejb nos permitirá hacer transacciones con la base de datos, es decir, nos ayudará hacer fácilmente los CRUDS respectivos con las diferentes entidades presentes dentro del proyecto.

Para lo dicho anteriormente se creó un GenericDAO global en donde solo debemos especificar la entidad y automáticamente la clase creada heredara todos los métodos necesarios para hacer un CRUD rápidamente.

La figura siguiente muestra cómo se crea un Ejb.

```
package com.map.services;

import java.util.ArrayList;

@Stateless
@LocalBean
public class MessageEjb extends GenericDAOImpl<Message, Integer>{

    public MessageEjb(){

    }


}
```

Ilustración V-44: CRUD mendiante GenericDAO

Elaborado por: Diego Ponce – Juan Prado

Se deben considerar los siguientes puntos:

- La clase se creará con UppearCase.
- La clase debe contener @Stateless y @LocalBean

- Debe extender la clase GenericDaoImpl especificando la entidad con la que tiene relación y el tipo de dato de su primary key.

5.4.2.3.2. Bean

Para hacer una relación entre la vista y los Ejb es necesario crear un Bean el mismo que se muestra en la siguiente figura para su correcta creación.

```
package com.map.beans;

import java.io.Serializable;

@ManagedBean(name="userBean")
@SessionScoped
public class UserBean implements Serializable{

    /**
     *
     */
    private static final long serialVersionUID = -7927820995066852655L;


    private User user = new User();
    private User userSearch = new User();
    private String passwordWSSH1 = new String();
    private String passwordWSSH2 = new String();
    private SHA sha = new SHA();
    private Calendar cal = Calendar.getInstance();
    private UserRole userRole = new UserRole();

    private String regexText = new String();

    private List<Role> roleList = new ArrayList<Role>();
    private List<User> userList = new ArrayList<User>();

    private Boolean isNew = new Boolean(Boolean.TRUE);

    @EJB
    UserEjb userAction;

    @EJB
    RoleEjb roleAction;
```

Ilustración V-45: Archivo Bean listo para la vinculación con su respectivo EJB

Elaborado por: Diego Ponce – Juan Prado

Consideraciones:

- El nombre de la clase se creará con el formato UppearCase.
- Se especificará el nombre del bean para que interactúe con la vista a través de la sentencia @ManagedBean.
- Se debe poner el tipo de sesión sea @SessionScoped como muestra la figura o ViewScoped, ApplicationScoped, etc.
- Se implementará Serializable para que la clase sea serializada.
- Se declara variable @EJB para realizar transacciones con la base de datos a través del ejb creado.

5.4.2.3.3. Vista

Las vistas se crearán en formato .xhtml. La figura siguiente muestra cómo se las realiza.

```
<h:form id="userForm">
<div class="Container100">
    <div class="ContainerIndent">
        <div class="Card ShadowEffect  ">
                <div class="ContainerIndent">
                    <h2 class="BigTopic">Búsqueda</h2>

                    <div class="Container60 Responsive50">
                    <p:panelGrid columns="2" layout="grid" style="border:0px !important; background:none;" styleClass="FormCon
                        <p:inputText styleClass="Wid90" placeholder="Nombre" value="#{userBean.userSearch.usrName}"/>
                        <p:inputText styleClass="Wid90" placeholder="Apellido" value="#{userBean.userSearch.usrLastname}"/>

                    </p:panelGrid>
                    </div>
                    <div class="Container40 Responsive50">
                    <p:panelGrid columns="3" layout="grid" style="border:0px !important; background:none;" styleClass="FormCon
                         <p:commandButton value="Buscar"  styleClass="PurpleButton" actionListener="#{userBean.search()}"/>
                         <p:commandButton value="Limpiar" styleClass="PurpleButton" actionListener="#{userBean.clean()}"/>
                         <p:commandButton icon="fa fa-child Fs16 White" styleClass="BlueTextButton RaisedButton" onclick="PF(
                    </p:panelGrid>
                    </div>
                </div>

        </div>
    </div>
</div>

<div class="Container100">
    <div class="ContainerIndent">
        <div class="Card ShadowEffect">
            <p:dataTable id="userTable" var="user" value="#{userBean.userList}">

                <p:column headerText="Nombre">
```

Ilustración V-46: Vistas de Usuario en archivos XML

Elaborado por: Diego Ponce – Juan Prado

Considerar:

- En el proyecto ya está configurado un template con la estructura visual que debe tener.
- Es importante especificar el nombre del form en h:form.
- Para poder acceder a los atributos del bean creado anteriormente es necesario declarar un value=”#{nombre del bean.atributo}” para los diferentes componentes del form sean botones, datatables, inputs, etc. Para estos componentes como se detalló en el marco teórico del capítulo II se están utilizando los que muestran en primefaces showcase.

5.4.2.4. Codificación Monitorización Virtual Móvil

5.4.2.4.1. Estructura de la aplicación Android Monitorización Virtual Móvil

La aplicación Monitorización Virtual Móvil se la ha desarrollado bajo la estructura ya predefinida por Android, la misma que se observa en la Figura V-47.

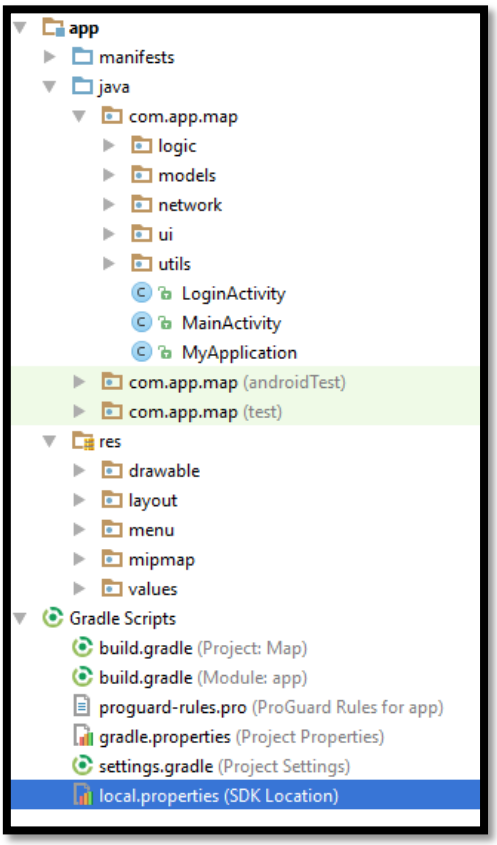


Ilustración V-47: Estructura de la aplicación Android de Monitorización Virtual Móvil

Elaborado por: Diego Ponce – Juan Prado

- **Directorio Manifest:** En este directorio se encuentra el archivo de configuración AndroidManifest.xml en el cual se definen todos los "activities" que se ejecutarán, "services", como también todos los permisos que tendrá la aplicación para su correcto funcionamiento.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.app.map">

    <!-- To auto-complete the email text field in the login form with the user's emails -->
    <uses-permission android:name="android.permission.GET_ACCOUNTS" />
    <uses-permission android:name="android.permission.READ_PROFILE" />
    <uses-permission android:name="android.permission.READ_CONTACTS" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="com.google.android.c2dm.permission.RECEIVE" />
    <uses-permission android:name="com.google.android.providers.gsf.permission.READ_GSERVICES" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
    />


    <application
        android:name=".MyApplication"
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Proyecto Mapa"
        android:supportsRtl="true"
        android:theme="@style/AppTheme"
        >
        <activity
            android:name=".LoginActivity"
            android:label="Sign in"
            android:launchMode="singleTop"
            android:windowSoftInputMode="adjustPan">
```

Ilustración V-48: Fichero "Manifest.xml"

Elaborado por: Diego Ponce – Juan Prado

- **Directorio Java:** Contiene el paquete principal de la aplicación la cual tiene la estructura siguiente:

- **Logic:** Son las clases que manejaran los CRUDS de las diferentes clases que necesitan ser guardadas dentro de una base de datos local dentro de Android

- **Models:** Son las entidades del negocio que se presentan dentro de la aplicación

- **Network:** Son las clases que permitirán el manejo de Google Cloud Messaging incluyendo mensajes downstream y upstream, la monitorización offline y online atraves de GPS, además se incluirán el manejo de web services.

- **UI:** Son todos los fragmentos y adapters que se manejan en las distintas vistas de la aplicación.

- **Utils**: Son las clases que se manejan distintas variables globales, manejo de sesión por usuario y estructuras de notificaciones como también parámetros necesarios para la estructura de envío de mensajes atraves de GCM. Cabe recalcar que estas clases son utilizadas frecuentemente en todos los paquetes antes mencionados.

- **Directorio res:** El directorio res contiene todos los recursos necesarios que usa la aplicación incluyendo imágenes, estilos, layouts, archivos de idiomas, etc. Encontramos las siguientes carpetas:

- **Drawable:** Contiene todas las imagines en distintos tamaños para que se ajusten al dispositivo en el cual se instale la aplicación, como también shapes que se usan en los layouts.

- **Layout:** Tiene todas las pantallas de la aplicación, incluyendo las de los fragmentos como la de los activities. Estas están en formato xml.

- **Values:** Contiene todas las carpetas con las variables que se usan dentro de las vistas incluyendo Strings, colores, textos.

- **Mipmap**: Este directorio contiene el logo de la aplicación.

- **Menú:** Definición del menú que se desplegará al ingresar a la aplicación en la parte superior de la aplicación la misma está en formato xml.

5.4.2.4.2. Construcción de Activities en la aplicación Monitorización Virtual Móvil

Los activities dentro de Android cumplen un rol muy importante, principalmente son las que interactúan con el usuario por lo tanto las mismas se

define como pantallas que forman una aplicación. Están compuestas por dos partes muy importantes las cuales son.

1. La clase es un archivo .java la misma que tiene como funcionalidad definir el comportamiento de los componentes de la vista y con otras vistas cuando el usuario interactúa con ella.

2. La parte gráfica es un archivo .xml que contiene todos los componentes de la vista.

5.4.2.4.2.1. Construcción de la parte lógica de una actividad

La figura siguiente muestra como es una clase en la aplicación Monitorización Virtual Móvil.

```
package com.app.map;

import ...

/**
 * A login screen that offers login via email/password.
 */
public class LoginActivity extends AppCompatActivity implements LoaderCallbacks<Cursor> {

    // UI references.
    private AutoCompleteTextView mEmailView;
    private EditText mPasswordView;
    private View mProgressView;
    private View mLoginFormView;
    private CallWebServices callWebServices;
    private BroadcastReceiver mRegistrationBroadcastReceiver;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

        if(MyApplication.getInstance().getPrefManager().getUser() !=null){
            startActivity(new Intent(this, MainActivity.class));
            finish();
        }
```

Ilustración V-49: Clase básica de la aplicación Móvil

Elaborado por: Diego Ponce – Juan Prado

Para crear un activity se debe considerar lo siguiente:

- Incluir el nombre del paquete al que pertenece la clase.
- Incluir imports dependiendo los elementos a los cuales se quiere acceder dentro de la clase.
- El nombre de la clase de ser con notación UpperCase.
- La clase debe extender Activity, AppCompatActivity dependiendo el comportamiento que debe tener el activity.
- La clase debe llevar un método OnCreate que permitirá crear el activity en sí.
- Se debe declarar un setContentView el cual hace referencia al layout con el cual va a interactuar el activity.

5.4.2.4.2.2. Construcción de la parte gráfica de una actividad.

La figura siguiente muestra como es la construcción de la parte gráfica de un activity la cual es un archivo .xml.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center_horizontal"
    android:orientation="vertical"
    android:paddingBottom="16dp"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp"
    tools:context="com.app.map.LoginActivity">

    <!-- Login progress -->
    <ProgressBar
        android:id="@+id/login_progress"
        style="?android:attr/progressBarStyleLarge"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="8dp"
        android:visibility="gone" />

    <ScrollView
        android:id="@+id/login_form"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <LinearLayout
            android:id="@+id/email_login_form"
            android:layout_width="match_parent"
```

Ilustración V-50: Base para la construcción de una Actividad

Elaborado por: Diego Ponce – Juan Prado

5.4.2.4.3. Construcción de Fragments en la aplicación Monitorización Virtual Móvil

Los “fragments” dentro del proyecto son igualmente una parte importante para la creación de las vistas ya que las mismas muestran el comportamiento de cierta parte de la interfaz dentro de un activity ya que los ciclos de vida de los mismos

dependen del activity anfitrión. De igual forma que el activity los fragmentos constan de dos partes importantes.

1. La clase es un archivo .java la misma que tiene como funcionalidad definir el comportamiento de los componentes de la vista y con otras vistas cuando el usuario interactúa con ella.

2. La parte gráfica es un archivo .xml que contiene todos los componentes de la vista.

5.4.2.4.4. Construcción de la parte lógica de una actividad

La figura siguiente muestra como es una clase en la aplicación Monitorización Virtual Móvil.

```
package com.app.map.ui.Fragments;

import ...

/**
 * Created by diego on 10/10/2016.
 */
public class ChatFragment extends Fragment implements ScreenShotable {

    private String TAG = ChatFragment.class.getSimpleName();



    @Nullable
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
        containerView = inflater.inflate(R.layout.content_chat_room, container, false);
        recyclerView = (RecyclerView) containerView.findViewById(R.id.recycler_view_chat);

        inputMessage = (EditText) containerView.findViewById(R.id.message);
        btnSend = (Button) containerView.findViewById(R.id.btn_send);

        messageArrayList = new ArrayList<>();
        messageArrayList = findMessages();
```

Ilustración V-51: Construcción de la parte Lógica de una Actividad

Elaborado por: Diego Ponce – Juan Prado

Para crear un fragment se debe considerar lo siguiente:

- Incluir el nombre del paquete al que pertenece la clase.
- Incluir imports dependiendo los elementos a los cuales se quiere acceder dentro de la clase.
- El nombre de la clase de ser con notación UpperCase.
- La clase debe extender Fragment.
- La clase debe llevar un método OnCreateView que permitirá crear el activity en sí.
- Se debe declarar un inflate el mismo que hará referencia a la vista que pertenece.

5.4.2.4.5. Construcción de la parte gráfica de una actividad

La figura siguiente muestra como es la construcción de la parte gráfica de un activity la cual es un archivo .xml.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:layout_behavior="android.support.design.widget.AppBarLayout$ScrollingVie..."
    android:paddingTop="70dp"
  >

    <android.support.v7.widget.RecyclerView
        android:id="@+id/recycler_view_chat"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:paddingBottom="20dp"
        android:scrollbars="vertical"
        android:layout_alignParentTop="true"
        android:layout_above="@+id/linearLayout" />

    <LinearLayout
        android:background="@android:color/white"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:orientation="horizontal"
        android:weightSum="4"
        android:id="@+id/linearLayout">
```

Ilustración V-52: Construcción de la parte gráfica de una Actividad

Elaborado por: Diego Ponce – Juan Prado

5.4.2.4.6. Creación de Modelo de Base de datos

La creación del modelo se la realiza como se muestra la siguiente imagen.

```
package com.app.map.models;

import ...

/**
 * Created by diego on 6/10/2016.
 */
public class Message extends RealmObject {

    private Integer id;
    private String name;
    private String message;
    private Date receivedDate;
    private Date sendDate;
    private Contact contact;
    private Boolean isSelf;
    private Boolean isSend;

    public Message(){

    }

    public Message(int id, Date receivedDate, Date sendDate, Contact contact, String message, Boolean isSelf){
        this.id = id;
        this.receivedDate = receivedDate;
        this.sendDate = sendDate;
        this.contact = contact;
        this.message = message;
        this.isSelf = isSelf;

    }
```

Ilustración V-53: Creación del modelo de Base de Datos

Elaborado por: Diego Ponce – Juan Prado

Para crear una clase para el modelo se debe considerar lo siguiente:

- Incluir el nombre del paquete al que pertenece la clase.
- Incluir imports dependiendo los elementos a los cuales se quiere acceder dentro de la clase.
- El nombre de la clase de ser con notación UpperCase.

- La clase debe extender RealmObject la cual sirve para que Realm (Base de datos para dispositivos móviles) permita saber que esa clase es el modelo en la cual se va realizar transacciones para crear, leer, eliminar y actualizar campos.

5.4.2.4.7. Creación de lógica para el modelo de la base de datos Realm.

La creación de la clase lógica para la gestión del modelo de la base de datos Realm se la muestra en la siguiente figura.

```
package com.app.map.logic;

import ...

/**...*/
public class MessageLogic {

    private Context context;
    RealmConfiguration realmConfig;
    Realm realm;

    public MessageLogic(Context context) { this.context = context; }

    public void persist(Message message){
        realmConfig = new RealmConfiguration.Builder(context).deleteRealmIfMigrationNeeded().build();
        Realm.setDefaultConfiguration(realmConfig);
        realm = Realm.getDefaultInstance();
        List<Message> messagesListDb = realm.where(Message.class).findAll();

        if (messagesListDb.size() == 0) {
            Message messageDb = new Message(1, message.getReceivedDate(), message.getSendDate(), message.getContact(), message.getMessage(),message.getSelf());
            realm.beginTransaction();
            Message realmContact = realm.copyToRealm(messageDb);
            realm.commitTransaction();
        } else {
            Message messageDb = new Message(getNextKey(), message.getReceivedDate(), message.getSendDate(), message.getContact(), message.getMessage(),message.getSelf());
            realm.beginTransaction();
            Message realmContact = realm.copyToRealm(messageDb);
            realm.commitTransaction();
        }
        realm.close();
    }
```

Ilustración V-54: Creación del modelo de datos para la aplicación móvil

Elaborado por: Diego Ponce – Juan Prado

Para crear una clase para la lógica se debe considerar lo siguiente:

- Incluir el nombre del paquete al que pertenece la clase.

- Incluir imports dependiendo los elementos a los cuales se quiere acceder dentro de la clase.
- El nombre de la clase debe ser con notación UpperCase.
- La clase tiene una función persist la misma que servirá para añadir un registro en la base de datos Realm, es importante considerar lo siguiente:

    - Hay que abrir la configuración y la instancia de Realm de la aplicación a través de RealConfiguration, y Realm.GetDefaultInstance.
    - Si se realizara alguna búsqueda general dentro de la base de datos se especifica con Realm.where.(“nombre del modelo”).findAll()
    - Si se inserta un registro dentro de la base de datos se debe empezar la transacción con realm.beginTransaction, luego se añade el registro con realm.copyToRealm(“Objeto a insertarse del modelo”), y se termina de insertar con un commit realm.commitTransaction();.

- Para que no haya problemas con la inserción de otros modelos de base de datos en las diferentes vistas es necesario cerrar la conexcion con Realm atraves de realm.close().

5.4.2.4.8. Consumo de Servicio Web

Para el consumo de web-services se está utilizando la librería que ofrece Android la cual se llama Google Volley.

A continuación, se muestra el método que permite hacer el consumo de un web service, la cual nos permitirá acceder desde cualquier parte de nuestro código a la respuesta del web service.

```
public void callRegister(final Context context, final String url, String email, String registerCode, String token){

    HashMap<String, String> params = new HashMap<~>();
    params.put(Config.URL_REGISTER_EMAIL, email);
    params.put(Config.URL_REGISTER_CODE, registerCode);
    params.put(Config.URL_REGISTER_TOKEN, token);

    JsonObjectRequest req = new JsonObjectRequest(url, new JSONObject(params),
            (response) -> {
                    try {
                        JSONObject jsonResponse = response;
                        if(!jsonResponse.getString("usrName").equals("null")){
                            User user = new User(jsonResponse.getString("usrId"),jsonResponse.getString("usrName"),jsonResponse.getString("usrLastname"),
                                    jsonResponse.getString("usrEmail"),jsonResponse.getString("usrNickname"));
                            MyApplication.getInstance().getPrefManager().storeUser(user);

                            Intent intent = new Intent(Config.LOGIN_RESPONSE);
                            intent.putExtra("loggin", "true");
                            LocalBroadcastManager.getInstance(context).sendBroadcast(intent);
                        }else
                            System.out.println("INGRESO FALLIDO");

                    } catch (JSONException e) {
                        System.out.println(e);
                        e.printStackTrace();
                    }
            }, (error) -> { VolleyLog.e("Error: ", error.getMessage()); });

    RequestQueue requestQueue = Volley.newRequestQueue(context);
    requestQueue.add(req);
}
```

Ilustración V-55: Clases de consumo de servicios Web

Elaborado por: Diego Ponce – Juan Prado

- Como podemos observar cada vez que se hace el consumo de webservice enviamos la respuesta atraves de un Broadcast.
- Para llamar al servicio es importante declarar un RequestQueue.Add(), el cual recibe como parámetro un JsonObjectRequest, este envía un Json al web service y por lo tanto recibe igualmente la respuesta en formato Json.

5.4.2.4.9. Consumo de GCM (Google Cloud Messaging)

El Consumo de Google Cloud Messaging se divide en cinco partes importantes las cuales se las explica a continuación.

5.4.2.4.9.1. Configuración Android Manifest

Es necesario especificar dentro del AndroidManifest que se crearan dos services los mismos tienen como objetivo realizar operaciones en segundo plano aunque la aplicación se cierre o se cambie a otra aplicación, estos services no están atados a una vista. Los dos services GcmIntentService y GcmServices permiten que las notificaciones y mensajes se reciban sin ninguna interrupción de otras actividades o procesos que se ejecuten dentro del dispositivo.

Por este motivo la siguiente figura muestra cómo se las debe configurar dentro del AndroidManifest.

```
<service
    android:name=".network.GCM.GcmIntentService"
    android:exported="false">
    <intent-filter>
        <action android:name="com.google.android.gms.iid.InstanceID" />
    </intent-filter>
</service>

<service
    android:name=".network.GCM.GcmService"
    android:exported="false">
    <intent-filter>
        <action android:name="com.google.android.c2dm.intent.RECEIVE" />
    </intent-filter>
</service>
```

Ilustración V-56: Archivo AndroidManifest.ml

Elaborado por: Diego Ponce – Juan Prado

5.4.2.4.9.2. Creación de GcmIntentService

Como podemos observar en el AndroidManifest es necesario que la clase GcmIntentService esté declarada como servicio, esta clase es sumamente importante ya que gracias a este servicio Google Cloud Messaging asignará un token único al dispositivo para que así tanto el servidor como los dispositivos sepan a que dispositivo se debe enviar el mensaje o la notificación.

```java
package com.app.map.network.GCM;

import ...

/**...*/
public class GcmIntentService extends IntentService {

    private void registerGCM(String email, String registerCode) {
        SharedPreferences sharedPreferences = PreferenceManager.getDefaultSharedPreferences(this);

        try {
            InstanceID instanceID = InstanceID.getInstance(this);
            Bundle extras = new Bundle();
            String token = instanceID.getToken(Config.GCM_SERVER_ID,
                    "GCM", extras);

            Log.e(TAG, "GCM Registration Token: " + token);

            // sending the registration id to our server
            sendRegistrationToServer(email,registerCode,token);

            sharedPreferences.edit().putBoolean(Config.SENT_TOKEN_TO_SERVER, true).apply();
        } catch (Exception e) {
            Log.e(TAG, "Failed to complete token refresh", e);

            sharedPreferences.edit().putBoolean(Config.SENT_TOKEN_TO_SERVER, false).apply();
        }
        // Notify UI that registration has completed, so the progress indicator can be hidden.
        Intent registrationComplete = new Intent(Config.REGISTRATION_COMPLETE);
        LocalBroadcastManager.getInstance(this).sendBroadcast(registrationComplete);
    }
```

Ilustración V-57: Creación de GCMIntentService

Elaborado por: Diego Ponce – Juan Prado

Es necesario tomar las siguientes consideraciones:

- Incluir el nombre del paquete al que pertenece la clase.
- Extender IntentService, esto nos ayudará a que el Servicio se lo llame desde cualquier parte sin necesidad de atarlo a un activity.
- Incluir imports dependiendo los elementos a los cuales se quiere acceder dentro de la clase.
- El nombre de la clase debe ser con notación UpperCase.
- La forma de asignar un token es con la siguiente sentencia que se muestra en la siguiente figura, donde llamamos a la instancia del celular y asignamos un token atraves de getToken("id del Servidor de GCM",

"GCM que especifica el servicio al que estamos pidiendo respuesta", "Bundle si pedimos un tiempo de vida del token").

```
InstanceID instanceID = InstanceID.getInstance(this);
Bundle extras = new Bundle();
String token = instanceID.getToken(Config.GCM_SERVER_ID,
        "GCM", extras);
```

Ilustración V-58: Gestión de vida del "Token"

Elaborado por: Diego Ponce – Juan Prado

5.4.2.4.9.3. Creación GcmService

Igualmente, este servicio como mencionamos anteriormente es necesario que esté especificado en el AndroidManifest.

El servicio GcmService permite a la aplicación estar pendiente a cualquier notificación o mensaje que sea enviado desde cualquier dispositivo o servidor, y así los mensajes sean almacenados en la base de datos local y muestre una notificación, aunque la aplicación se esté ejecutando en segundo plano o si es el caso muestre en pantalla si la aplicación este abierta.

La figura siguiente muestra cómo se ha implementado esta clase y servicio.

```
package com.app.map.network.GCM;

import ...

public class GcmService extends GcmListenerService {

    public GcmService() { logger = new LoggingService.Logger(this); }

    @Override
    public void onMessageReceived(String from, Bundle data) {
        sendNotification("Received GCM Message: " + data.toString());
    }

    @Override
    public void onDeletedMessages() { sendNotification("Deleted messages on server"); }

    @Override
    public void onMessageSent(String msgId) {
        sendNotification("Upstream message sent. Id=" + msgId);
    }

    @Override
    public void onSendError(String msgId, String error) {
        sendNotification("Upstream message send error. Id=" + msgId + ", error" + error);
    }


    private void processUserMessage(String title, boolean isBackground, String msg,
                                    String username, String response,String lastname, String nickname) {
        logger.log(Log.INFO, msg);
    }
```

Ilustración V-59: Creación del GCMService

Elaborado por: Diego Ponce – Juan Prado

Es necesario tomar las siguientes consideraciones:

- Incluir el nombre del paquete al que pertenece la clase.
- Extender GcmListenerService, a que el servicio esté pendiente de cualquier mensaje o notificación que llegue al dispositivo.
- Incluir imports dependiendo los elementos a los cuales se quiere acceder dentro de la clase.
- El nombre de la clase debe ser con notación UpperCase.
- Implementar métodos heredados de GcmIntentService los cuales son:

- **OnMessageReceived:** Este método nos permite descomprimir a través de un bundle todo el mensaje que ha sido enviado al dispositivo y quien fue el que lo envío.

- **OnDeleteMessages:** Ayuda a saltar un mensaje si los mensajes fueron borrados del servidor GCM.

- **OnMessageSent:** Si el mensaje fue enviado correctamente desde el servidor GCM al dispositivo destino muestra un mensaje de que fue enviado correctamente.

- **OnSentError:** Si el mensaje no pudo llegar correctamente al Servidor o el Servidor no envió el mensaje al dispositivo por razones como que el Id del servidor es incorrecto, o no hay ningún dispositivo con el token correcto, GCM notifica a través de este método.

- En nuestro caso hemos creado un método llamado processUserMessage que nos permitirá guardar todo el mensaje por usuario en la base de datos local, además de poder mostrar notificaciones si el aplicativo se está ejecutando en segundo plano.

5.4.2.4.9.4. Creación GcmUpstream

Los mensajes upstream se definen como aquellos que son enviados de dispositivo móvil a servidor de aplicaciones para que estos se muestren en la aplicación web. Para realizar este proceso se ha definido una clase que hace este trabajo formando toda la data a través de un Bundle.

La figura siguiente muestra cómo se forma el Bundle y como se envía el mensaje con GCM.

```
package com.app.map.network.GCM;

import ...

/**
 * Created by diego on 7/11/2016.
 */
public class GcmUpstream {

    public GcmUpstream() { }

    public void doGcmSendUpstreamMessage(Activity activity, String senderId, String msgId, String type, String message, String nickname)
        final GoogleCloudMessaging gcm = GoogleCloudMessaging.getInstance(activity);
        final String senderID = senderId;
        final String msgID = msgId;
        final String msgIdentificator = msgId;
        final Bundle data = new Bundle();
        SimpleDateFormat format = new SimpleDateFormat("yyyy-MM-dd hh:mm:ss");

        data.putString("message_type", type);
        data.putString("sentDate", format.format(Calendar.getInstance().getTime()));
        data.putString("messageText", message);
        data.putString("nickname", nickname);

        new AsyncTask<Void, Void, String>() {
            @Override
            protected String doInBackground(Void... params) {
                try {
                    // gcm.send(senderID + "@gcm.googleapis.com", msgID, data);
                    gcm.send(senderID + "@gcm.googleapis.com", msgID, Long.parseLong("1000"), data);
```

Ilustración V-60:Creación del GCMUpstream

Elaborado por: Diego Ponce – Juan Prado

Es necesario tomar las siguientes consideraciones:

- Incluir el nombre del paquete al que pertenece la clase.

- Incluir imports dependiendo los elementos a los cuales se quiere acceder dentro de la clase.
- El nombre de la clase debe ser con notación UpperCase.
- Se ha definido un método que recibirá como parámetro el SenderId del servidor como también el mensaje y el usuario quien lo envía.
- Para poder enviar el mensaje hay que declarar una variable que nos permita utilizar el método para envío por upstream el cual es GoogleCloudMessaging, esta variable nos permitirá utilizar el método send("SenderID+@gcm.googleapis.com","MessageID","TTL", "Bundle que especifica la información a ser enviada").

5.4.2.4.9.5. Creación GcmDownstream

Los mensajes downstream se definen como aquellos que son enviados de dispositivo móvil a dispositivo móvil. Para realizar este proceso se ha definido una clase que hace este trabajo formando toda la data.

La figura siguiente muestra cómo se forma el mensaje y como se envía el mensaje con GCM.

```java
package com.app.map.network.GCM;
import ...

/**...*/
public class GcmDownstream {

    public GcmDownstream(){}

    public void doGcmSendDowstreamMessage(final Activity activity, final String senderId, final String registrationId, String msgId, St

        final MessageGcmStructure.Builder messageBuilder = new MessageGcmStructure.Builder();
        //...
        messageBuilder.addData("Message",message);
        messageBuilder.addData("UserName",userName);
        messageBuilder.addData("UserLastname",userLastname);
        messageBuilder.addData("UserNickname",userNickname);
        messageBuilder.addData("Response","true");
        messageBuilder.addData("Device","MOVIL");
        messageBuilder.addData("content-available","1");
        messageBuilder.addData("to",senderId);
        messageBuilder.addData("priority","high");
        messageBuilder.notificationSound("default");
        messageBuilder.notificationBody(message);
        messageBuilder.notificationTitle(userName+" "+userLastname);
        messageBuilder.notificationTag("1");

        new AsyncTask<Void, Void, String>() {
            @Override
            protected String doInBackground(Void... params) {
                GcmSenderSideSender sender = new GcmSenderSideSender(senderId);
                try {
                    sender.sendHttpJsonDownstreamMessage(registrationId, messageBuilder.build());
```

Ilustración V-61: Creación del GCMDownStream

Elaborado por: Diego Ponce – Juan Prado

Es necesario tomar las siguientes consideraciones:

- Incluir el nombre del paquete al que pertenece la clase.
- Incluir imports dependiendo los elementos a los cuales se quiere acceder dentro de la clase.
- El nombre de la clase debe ser con notación UpperCase.
- Se ha definido un método que recibirá como parámetro el SenderId del servidor, el registrationId que es el token del dispositivo al que se enviará el mensaje el mensaje y el usuario quien lo envía.
- Para poder enviar el mensaje hay que formarlo con un Map el mismo se lo construye dentro de una clase llamada MessageGcmStructure y luego se lo envía a través de GcmSenderSideServer.

sendHttpJsonDownstreamMessage(“token del dispositivo”,”Mensaje construido a partir de MessageGcmStructure”).

5.5. PROTOTIPO FINAL

Una vez terminada la fase de implementación del proyecto de software queda definida las interfaces de usuario de la siguiente manera.

5.5.1. Interfaces de Usuario Plataforma Web

5.5.1.1. Pantalla de Inicio de Sesión

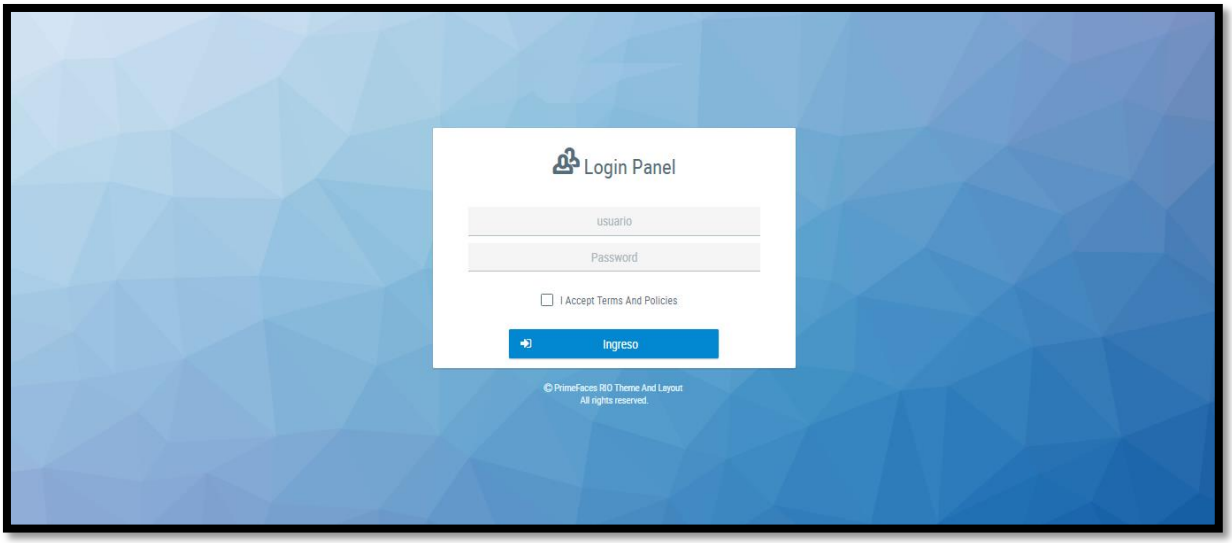


Ilustración V-62: Interfaz final de usuario: Inicio de sesión de aplicación Web

Elaborado por: Diego Ponce – Juan Prado

5.1.1.1. Pantalla Principal – Lista de Usuarios

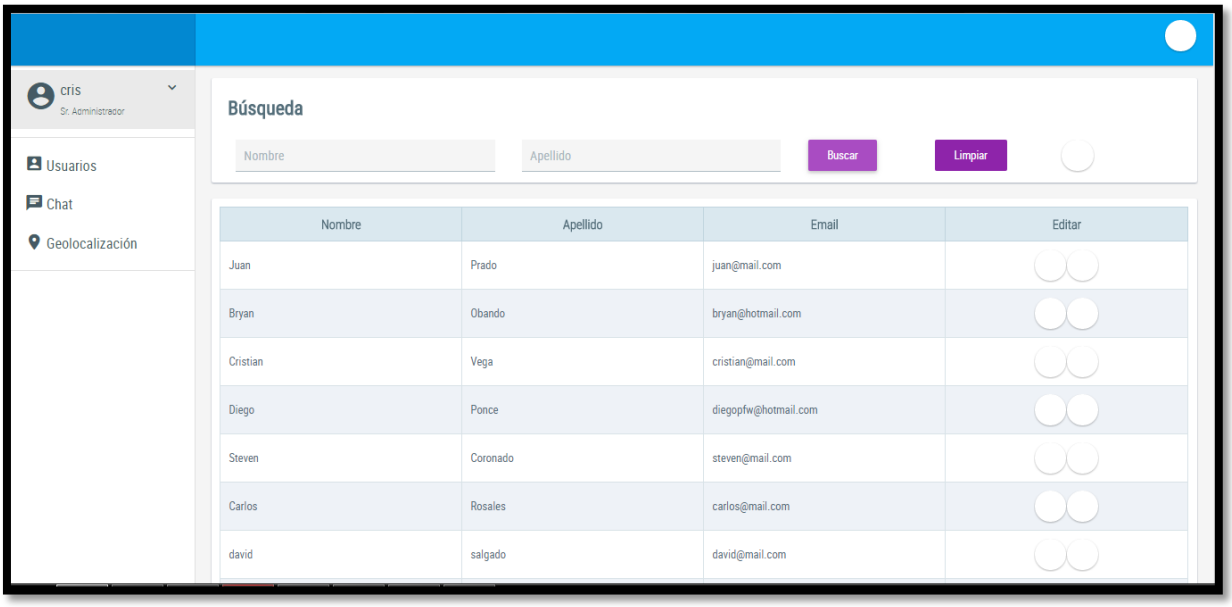


Ilustración V-63: Interfaz final de usuario: Pantalla Principal – Lista de Usuarios de Aplicación Web

Elaborado por: Diego Ponce – Juan Prado

5.1.1.2. Pantalla de Mensajería Instantánea

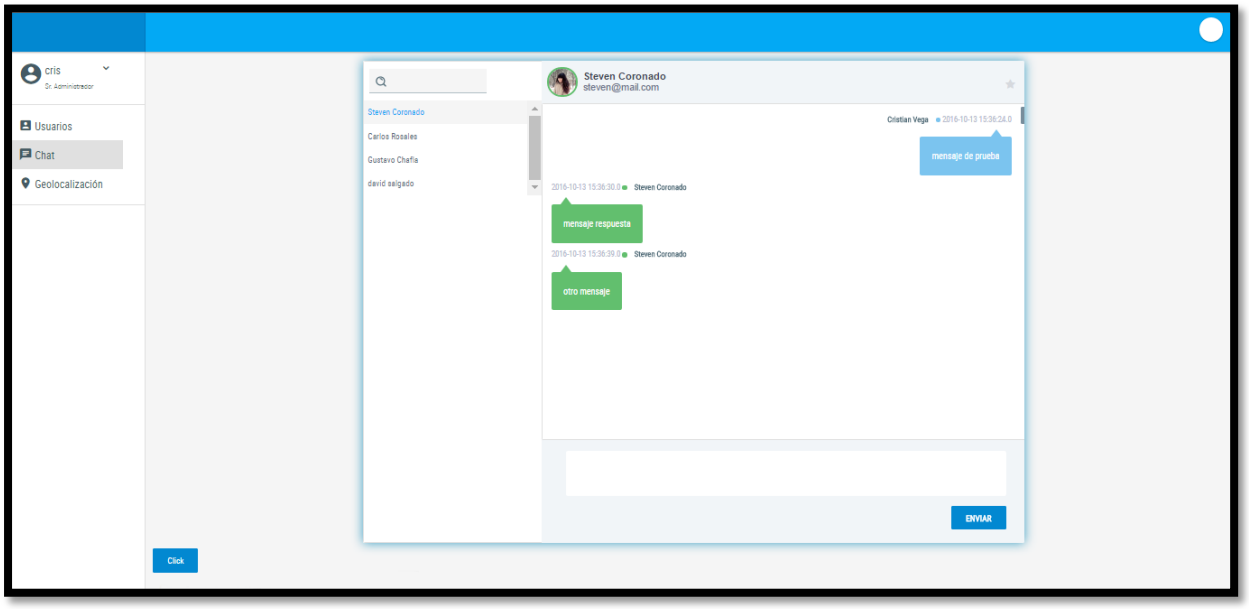


Ilustración V-64: Interfaz final de usuario: Pantalla de mensajería instantánea de Aplicación Web

Elaborado por: Diego Ponce – Juan Prado

5.1.1.3. Pantalla de monitor GPS y Mapas

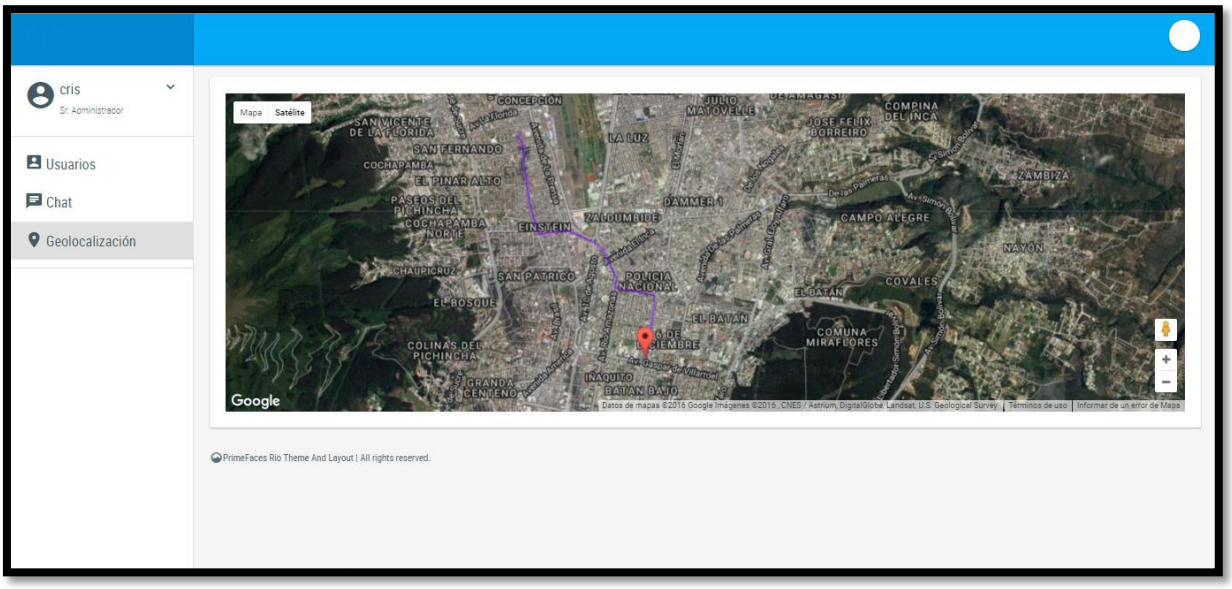


Ilustración V-65: Interfaz final de usuario: Pantalla de monitoreo GPS y Mapas de Aplicación Web

Elaborado por: Diego Ponce – Juan Prado

5.1.2. Interfaces de Usuario Dispositivo Móvil

5.1.2.1. Pantalla de Registro de Usuario

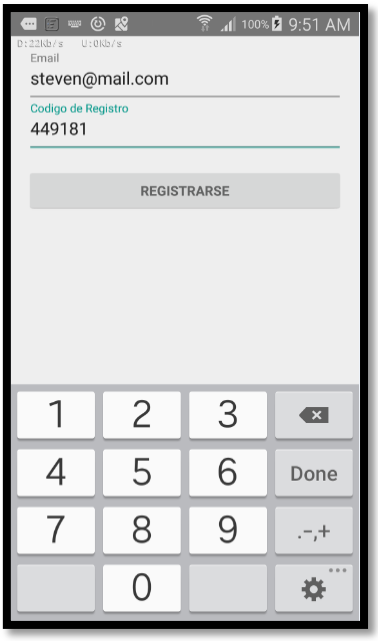


Ilustración V-66: Interfaz final de usuario: Pantalla inicial de registro de Aplicación Móvil

Elaborado por: Diego Ponce – Juan Prado

5.1.2.2. Pantalla de Menú Principal

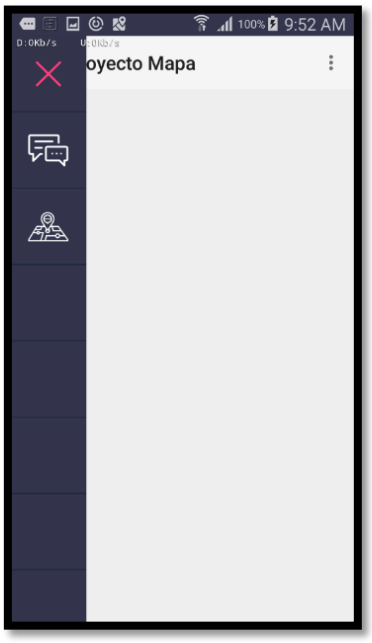


Ilustración V-67: Interfaz final de usuario: Fragmento de menú principal de Aplicación Móvil

Elaborado por: Diego Ponce – Juan Prado

5.1.2.3. Pantalla de Contactos

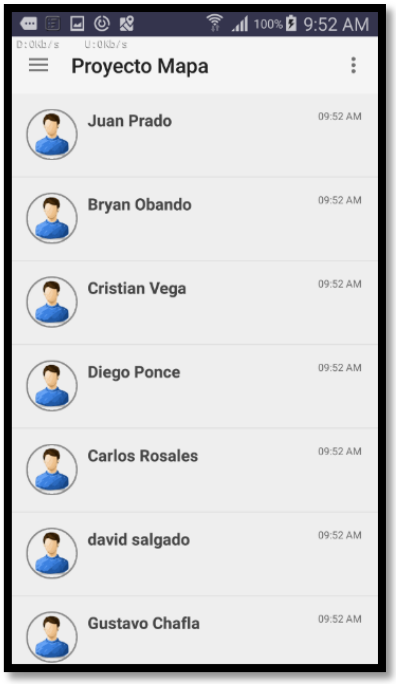


Ilustración V-68: Interfaz final de usuario: Pantalla de usuarios registrados de Aplicación Móvil

Elaborado por: Diego Ponce – Juan Prado

5.1.2.4. Pantalla de Mensajería Instantánea

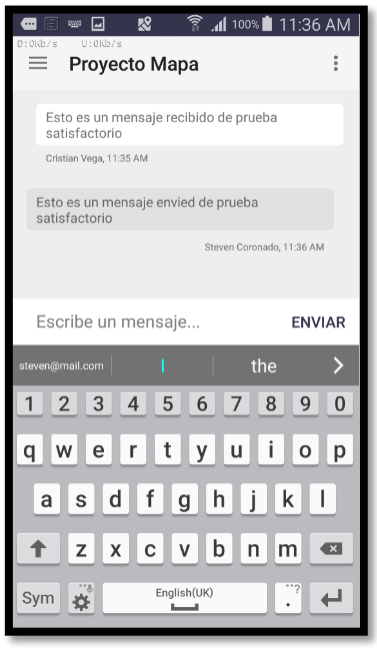


Ilustración V-69: Interfaz final de usuario: Pantalla de mensajería Instantánea de Aplicación Móvil

Elaborado por: Diego Ponce – Juan Prado

5.1.2.5. Pantalla de Geo-Posicionamiento y Mapas

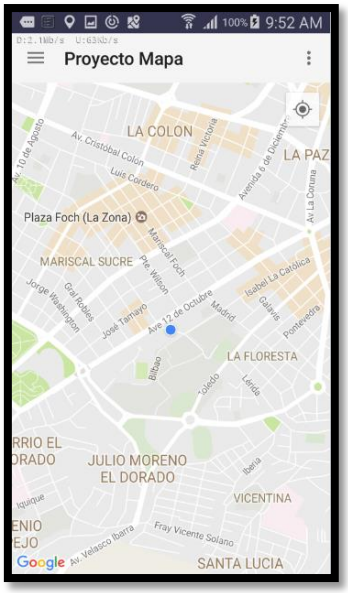


Ilustración V-70:Interfaz final de usuario: Pantalla de posición GPS y Mapas de Aplicación Móvil

Elaborado por: Diego Ponce – Juan Prado

5.1.2.6. Cuadro de opciones en mapas y contactos

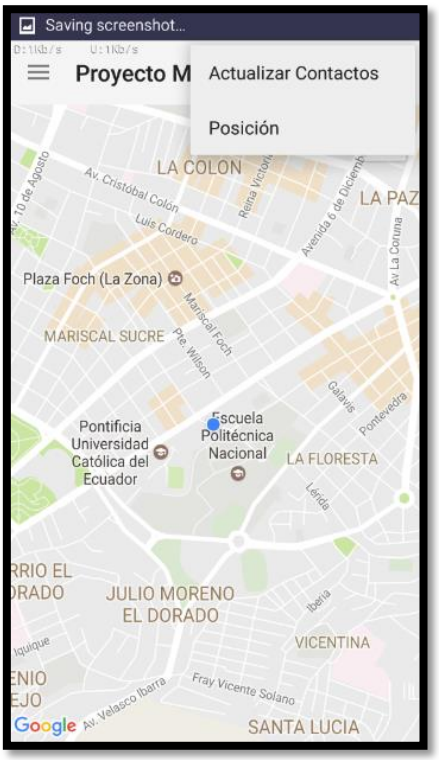


Ilustración V-71: Interfaz final de usuario: Cuadro de opciones de Aplicación Móvil

Elaborado por: Diego Ponce – Juan Prado

5.6. PRUEBAS

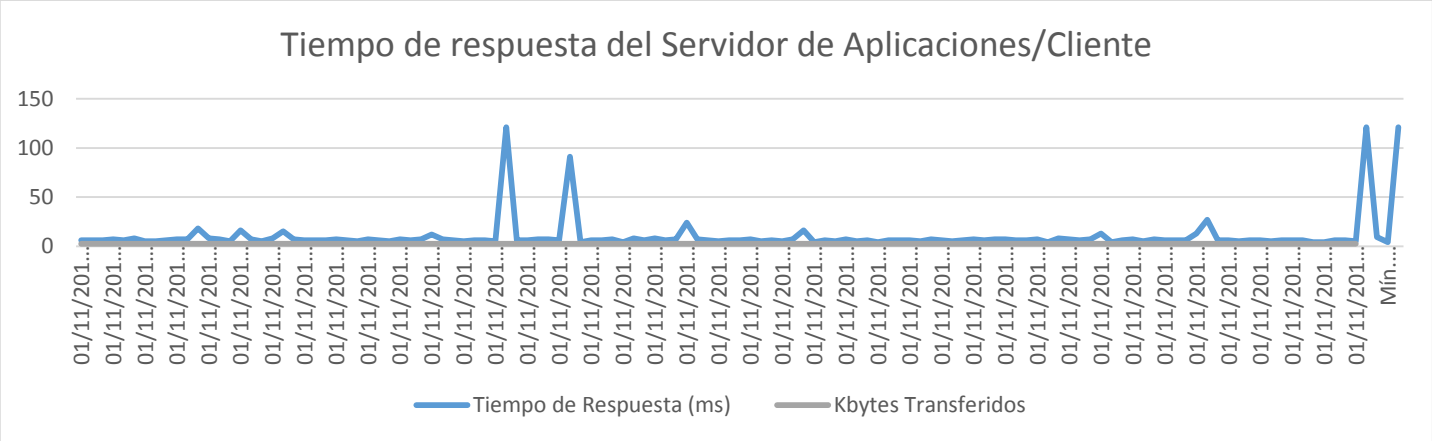
En el presente apartado se ha realizado el monitoreo y evaluación del funcionamiento tanto de la plataforma web como de la aplicación móvil, de los cuales se tomarán parámetros de medida en cuanto a la comunicación y transporte de datos, a continuación, serán analizados y descritos para posibles soluciones y usos para aplicaciones en entornos reales tanto domésticos como empresariales o corporativos.

5.6.1. Estudio de Pruebas – Plataforma Web

Para la recolección de pruebas se ha supervisado los siguientes parámetros dentro del funcionamiento de la Plataforma Web.

5.6.1.1. Tiempo de respuesta por parte del Servidor de Aplicaciones con respecto a un cliente

- **Descripción:** El siguiente estudio evaluará el tiempo que transcurre en la transferencia de datos o paquetes de información entre el Servidor de Aplicaciones y un cliente, independiente de que sea dispositivo móvil o estación web.



5.6.1.1.1. Periodo de Tiempo: 10 Horas

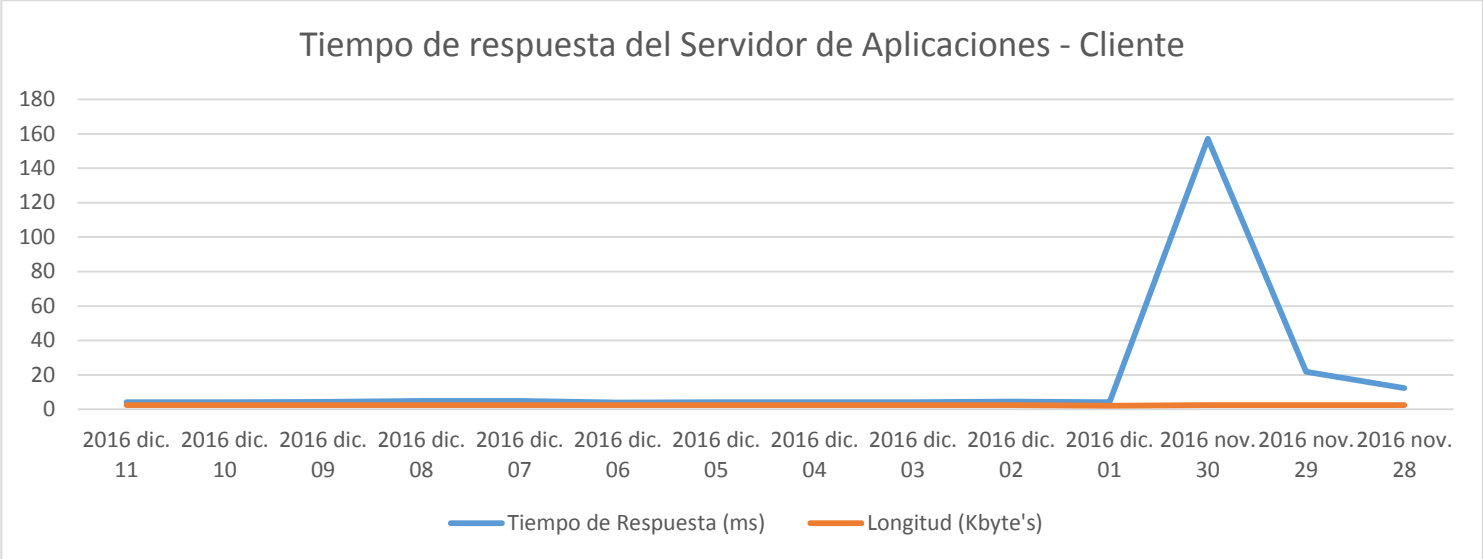
Elaborado por: Diego Ponce – Juan Prado

Ilustración V-72: Tiempo de respuesta del Servidor de Aplicaciones con respecto a un cliente determinado (10 hrs.)

- **Notas:** En base a los datos recogidos de la tabla 38 del apartado de Anexos, se estudiaron 121 transacciones de consulta de usuarios desde un cliente determinado al Servidor de Aplicaciones por un lapso de 10 horas, las transacciones fueron realizadas cada 5 min., las cuales el promedio de solicitud y entrega de paquetes es de 8,5 milisegundos (ms) transfiriendo en cada paquete una longitud de 2 Kbyte's.

Los paquetes que tuvieron un tiempo de respuesta entre 4 y 25 ms han tenido una mayor eficiencia con carga instantánea en la consulta de los usuarios.

Los paquetes con un tiempo de respuesta mayor de 25 ms han tenido problemas de conexión, o paquetes con envíos reincidentes por pérdida de datos.



5.6.1.1.2. Periodo de Tiempo: 14 Días

Ilustración V-73: Tiempo de respuesta del Servidor de Aplicaciones – Cliente (14 Días)

Elaborado por: Diego Ponce – Juan Prado

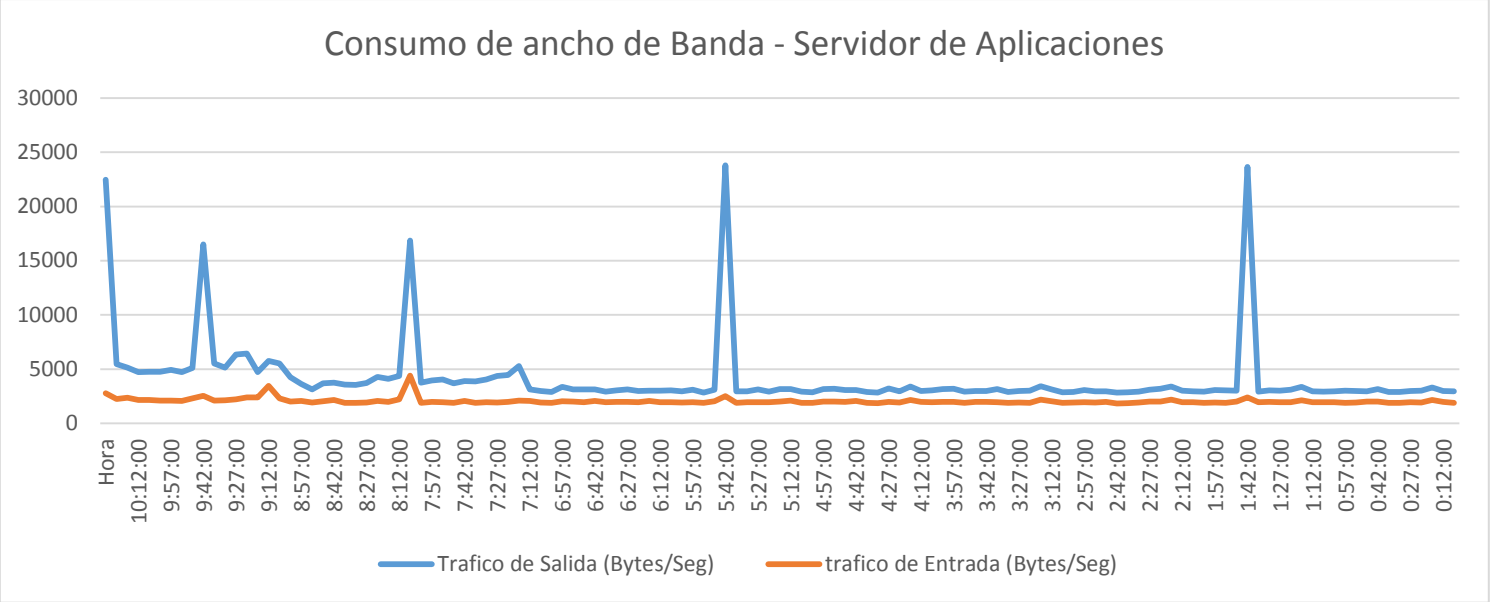
- **Notas:** En base a los datos obtenidos de la tabla 39 del apartado de Anexos, se estudiaron 14 transacciones las cuales representan a los días evaluados de consulta de usuarios desde un cliente determinado al Servidor de Aplicaciones, las transacciones fueron realizadas cada 24 horas, las cuales el promedio de solicitud y entrega de paquetes es de 17,5 milisegundos (ms) transfiriendo en cada paquete una longitud de 2,4 Kbyte's en promedio.

Los paquetes que tuvieron un tiempo de respuesta entre 4 y 25 ms han tenido una mayor eficiencia con carga instantánea en la consulta de los usuarios.

Los paquetes con un tiempo de respuesta mayor de 25 ms han tenido problemas de conexión, o paquetes con envíos reincidentes por pérdida de datos, tal como se puede observar el 30 de noviembre hubo problemas de conexión.

5.6.1.2. Evaluación de consumo de ancho de banda consumido por el Servidor de Aplicaciones

- **Descripción:** Este estudio evalúa el ancho de banda consumido tanto por el sistema operativo del servidor de aplicaciones como la aplicación web que la contiene.



5.6.1.2.1. Periodo de Tiempo: 10 Horas

Ilustración V-74: Consumo de ancho de Banda por parte del Servidor de Aplicaciones

Elaborado por: Diego Ponce – Juan Prado

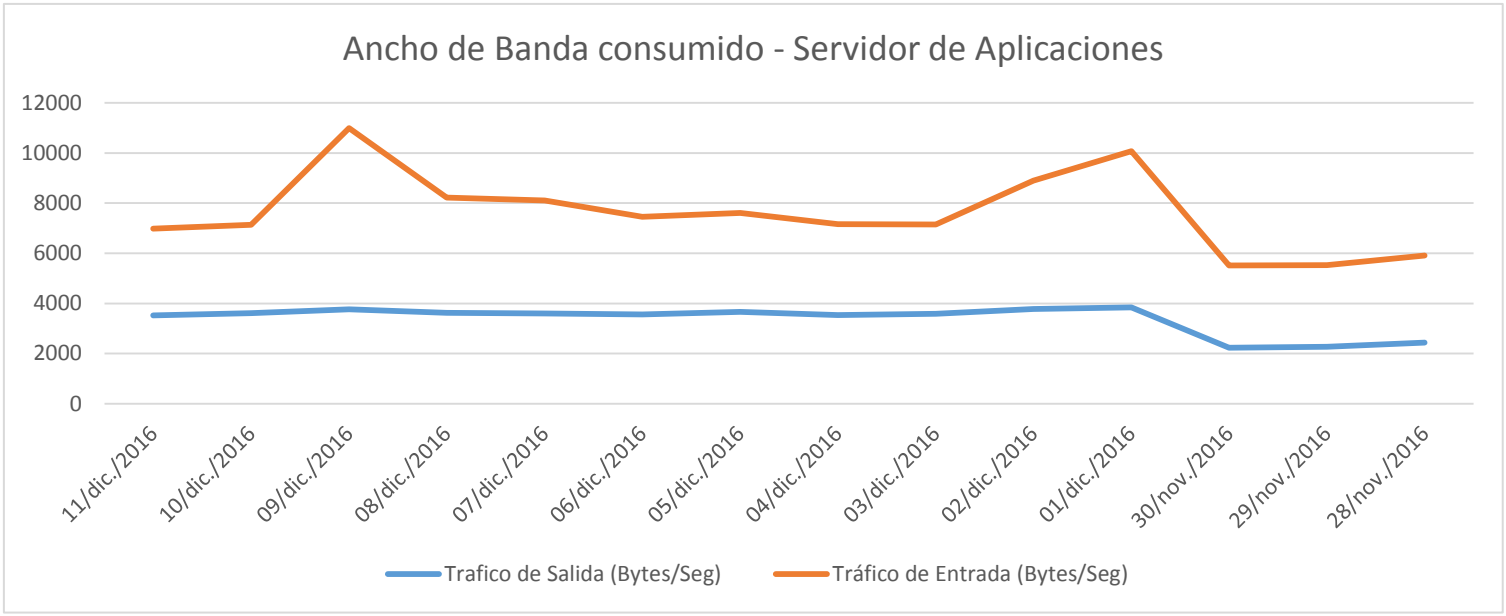
- **Notas:** En base a los datos obtenidos de la tabla 40 del apartado de Anexos, se analizan 125 transacciones que han reflejado el consumo de banda ancha con respecto a la descarga y carga de datos para el sistema operativo y las transacciones de la plataforma web.

Las transacciones fueron realizadas cada 5 minutos durante un período de 10 Horas, en el cual:

- El promedio de tráfico de salida es de 4163,34 Bytes/seg.
- El promedio de tráfico de entrada es de 2042,76 Bytes/seg.
- En el período evaluado el resultado mínimo de trafico de salida es de 2829,24 Bytes/seg.
- En el peíodo de evaluación el resultado mínimo de tráfico de entrada es de 2042,76 Bytes/seg.

- El valor máximo generado por los datos de salida es de 2819,24 Bytes/seg.
- El resultado máximo generado por el tráfico de entrada es de 4389,24 Bytes/Segundo

El valor máximo consumido es debido a descarga de archivos en el servicio de actualización del Sistema Operativo como tal.



5.6.1.2.2. Periodo de Tiempo: 14 Días.

Ilustración V-75: Consumo de ancho de banda por parte del Servidor de Aplicaciones

Elaborado por: Diego Ponce – Juan Prado

- **Notas:** En base a los datos obtenidos de la tabla 41 de Anexos, adicional, se analizan 14 transacciones que han reflejado los días de consumo de banda ancha con respecto a la descarga y carga de datos para el sistema operativo y las transacciones de la plataforma web.

Los datos fueron tomados cada 24 horas dando los siguientes resultados:

- El promedio de tráfico de salida es de 7623,47 Bytes/seg.
- El promedio de tráfico de entrada es de 3356,56 Bytes/seg.
- En el período evaluado el resultado mínimo de trafico de salida es de 5508,83 Bytes/seg.
- En el período de evaluación el resultado mínimo de tráfico de entrada es de 2228,17 Bytes/seg.
- El valor máximo generado por los datos de salida es de 10997,25 Bytes/seg.
- El resultado máximo generado por el tráfico de entrada es de 3836,38 Bytes/seg.

El valor máximo consumido es debido a descarga de archivos en el servicio de actualización del Sistema Operativo como tal.

5.6.1.3. Evaluación del consumo de memoria RAM en el servidor de Aplicaciones

5.6.1.3.1. Periodo de Tiempo: 14 Días.

- **Descripción:** En el presente análisis se recoge los datos que reflejan la memoria RAM consumida por el Servidor de Aplicaciones.

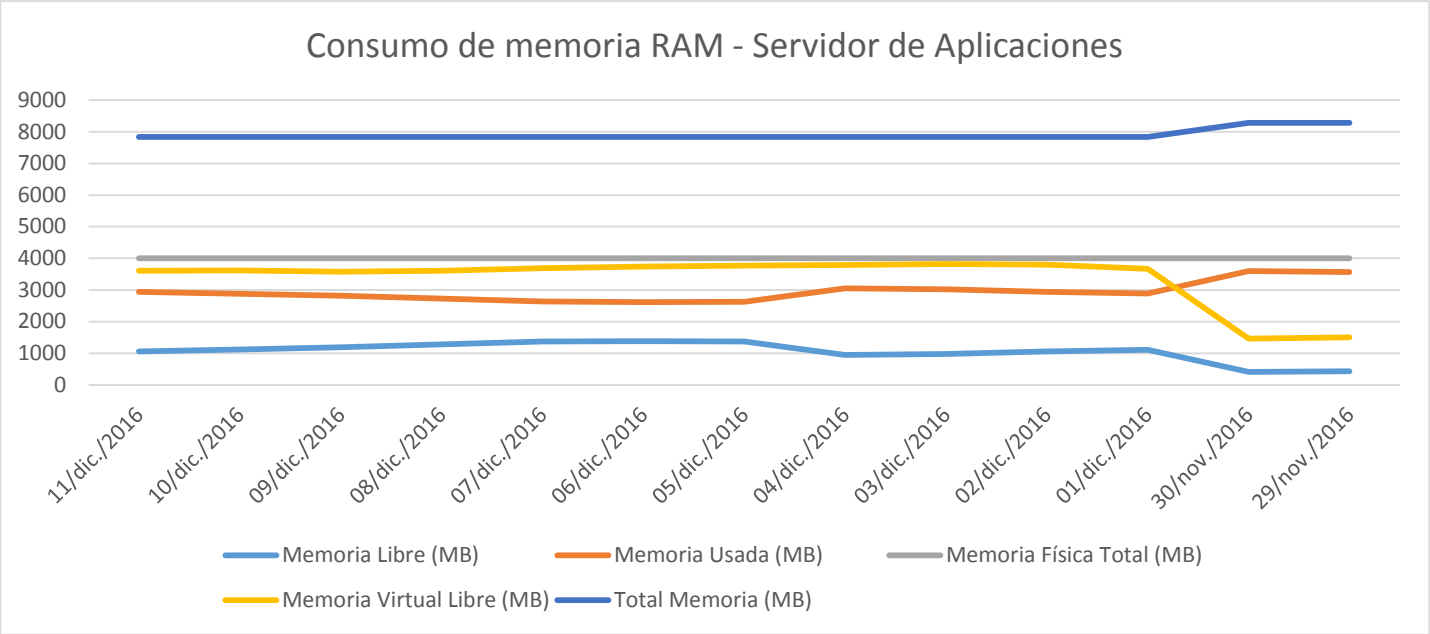


Ilustración V-76: Consumo de memoria RAM por parte del Servidor de Aplicaciones

Elaborado por: Diego Ponce – Juan Prado

- **Notas:** En base a los datos obtenidos de la tabla 42 de Anexos, se analizan 13 transacciones que han reflejado los días de consumo de memoria RAM con respecto al funcionamiento del Sistema Operativo y sus servicios, adicional el motor del servidor de aplicaciones y la plataforma web.

Los datos fueron tomados cada 24 horas dando los siguientes resultados, tomando en cuenta que el equipo tiene 4 GB de memoria RAM, y una memoria virtual del mismo valor.

- El promedio de uso de los módulos físicos de memoria fue de 2946,60 MBytes.

- El valor máximo de memoria usada fue de 3593,97 MBytes.
- El valor mínimo de memoria usada fue de 2616,28 MBytes.

La irregularidad del consumo de RAM es debido tanto a las solicitudes y transacciones de la aplicación web como de las tareas varias y extras del Sistema Operativo como tal.

5.6.2. Estudio de Pruebas – Aplicación Móvil

5.6.2.1. Evaluación del consumo datos en el envío de mensajes por parte del Dispositivo Móvil

- **Descripción:** En el presente análisis se recoge los datos que el dispositivo móvil consume al enviar mensajes de texto a otro dispositivo cliente o administrador.

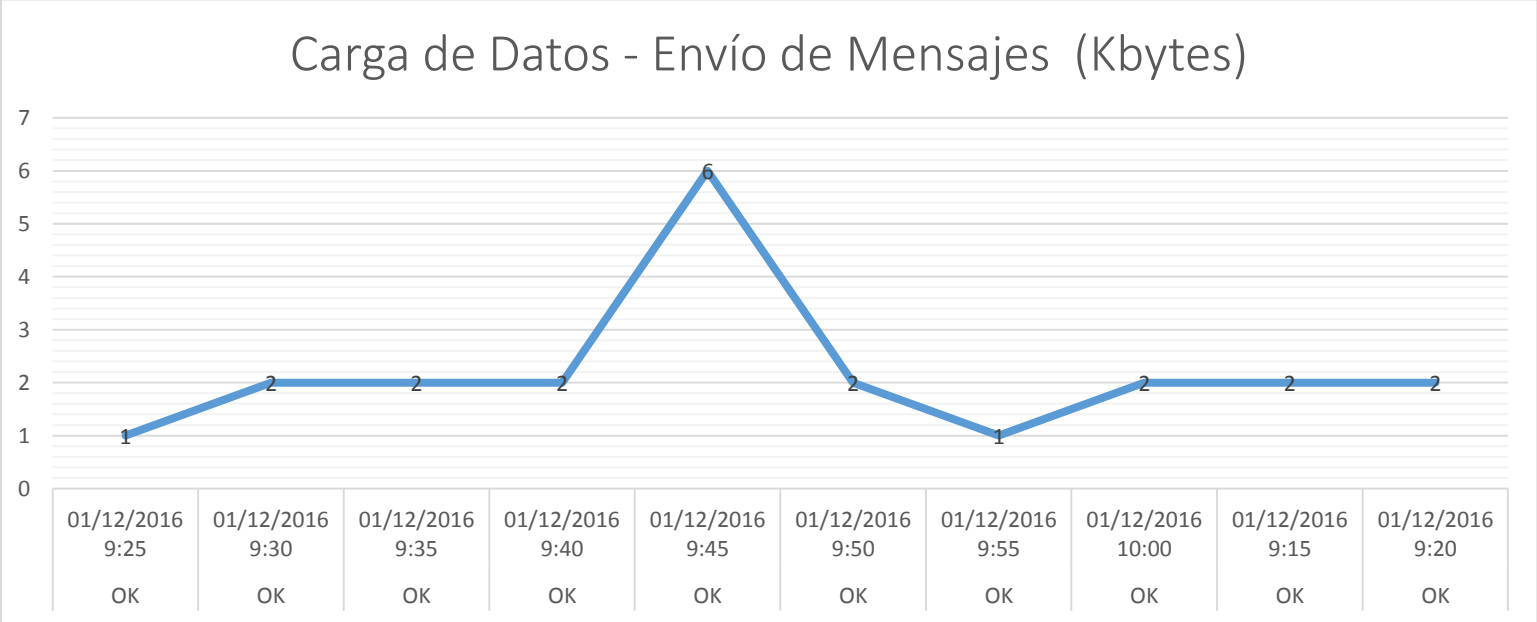


Ilustración V-77: Carga de Datos en el envío de mensajes por parte del Dispositivo Móvil

Elaborado por: Diego Ponce – Juan Prado

- **Notas:** En base a los datos obtenidos de la tabla 43 de Anexos, se analizan 10 transacciones que refleja el consumo de datos (Kbyte's) para poder enviar mensajes de texto a otros dispositivos.

Los datos fueron tomados cada 5 minutos dando los siguientes resultados, tomando en cuenta que el dispositivo uso una conexión Wi-Fi:

- El promedio de los datos utilizados fue de 2,2 Kbyte's en el lapso de 1 hora.
- El mensaje que tuvo menor consumo de datos fue de 1 Kbyte.
- El mensaje que tuvo mayor consumo de datos fue de 6 Kbyte's.

Los mensajes han sido enviados tomando un estándar de 20 a 75 caracteres.

5.6.2.2. Evaluación del consumo datos móviles en la recepción de mensajes por parte del Dispositivo Móvil

- **Descripción:** Este estudio recolecta el consumo de datos a la recepción de mensajes de texto por parte del Dispositivo Móvil.

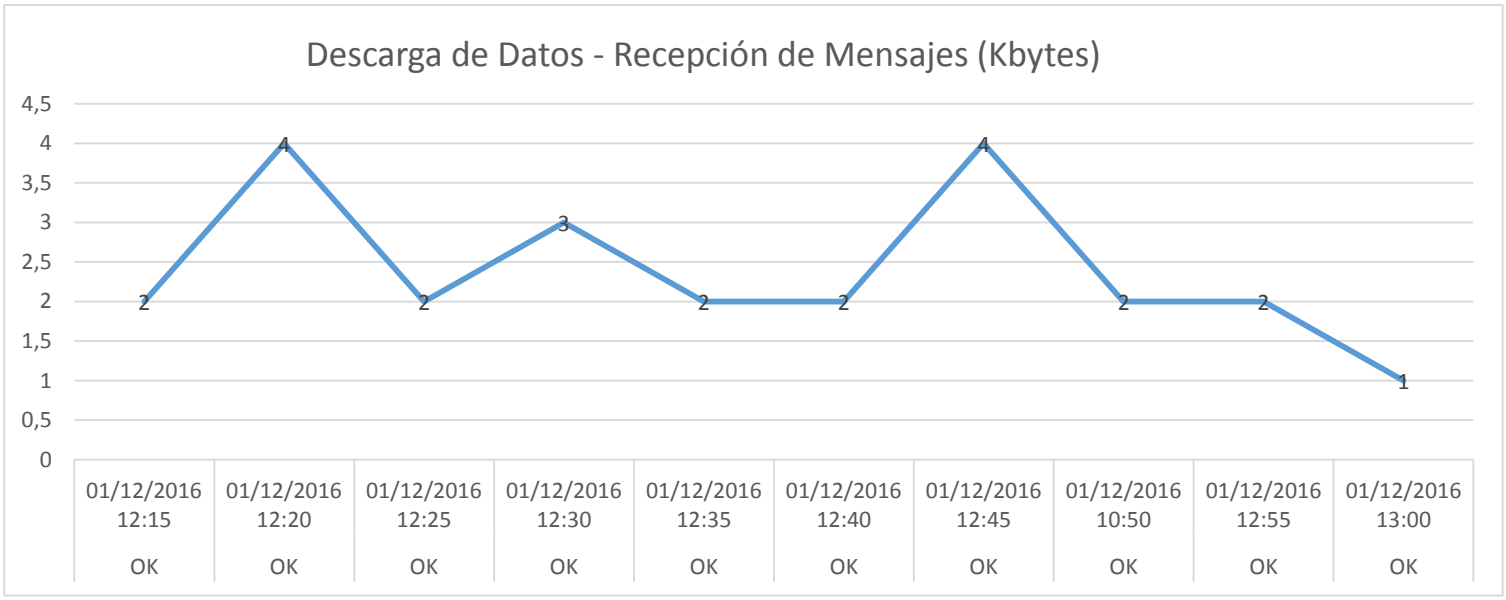


Ilustración V-78: Descarga de Datos en la recepción de mensajes por parte del Dispositivo Móvil

Elaborado por: Diego Ponce – Juan Prado

- **Notas:** En base a los datos obtenidos de la tabla 44 de Anexos, se analizan 10 transacciones que refleja el consumo de datos (Kbyte's) para poder receptar mensajes de texto de otros dispositivos clientes.

Los datos fueron tomados cada 5 minutos dando los siguientes resultados, tomando en cuenta que el dispositivo uso una conexión WI-FI:

- El promedio de los datos utilizados fue de 2,4 Kbyte’s en el lapso de 1 hora.
- El mensaje recibido que tuvo menor consumo de datos fue de 1 Kbyte.
- El mensaje recibido que tuvo mayor consumo de datos fue de 4 Kbyte’s.

Los mensajes han sido receptados tomando un estándar de 20 a 75 caracteres.

5.6.2.3. Evaluación del consumo datos en el envío de posición GPS por parte del Dispositivo Móvil

- **Descripción:** El presente análisis recoge información sobre los datos GPS que envía el Dispositivo Móvil.

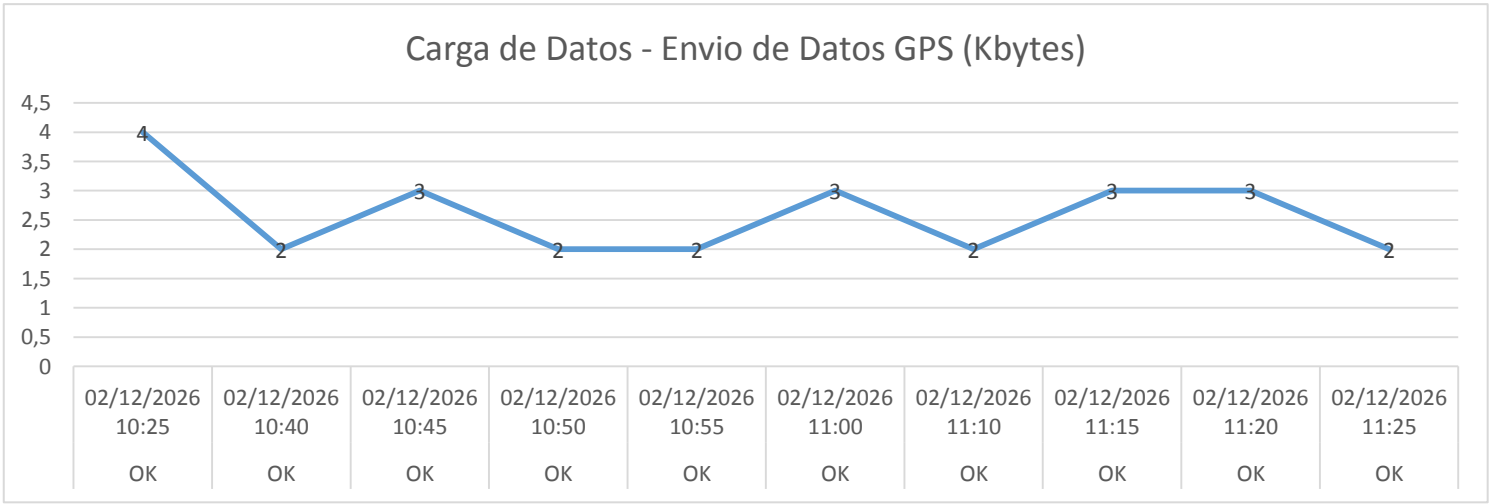


Ilustración V-79: Carga de Datos en el envío de ubicación GPS por parte del Dispositivo Móvil

Elaborado por: Diego Ponce – Juan Prado

- **Notas:** En base a los datos obtenidos de la tabla 45 de Anexos, se analizan 10 transacciones que refleja el consumo de datos (Kbyte's) para que el dispositivo móvil pueda notificar su posición al administrador.

Los datos fueron tomados cada 5 minutos dando los siguientes resultados, tomando en cuenta que el dispositivo uso una conexión Wi-Fi:

- El promedio de los datos utilizados fue de 2,6 Kbyte's en el lapso de 1 hora.
- El mensaje que tuvo menor consumo de datos fue de 2 Kbyte.
- El mensaje que tuvo mayor consumo de datos fue de 4 Kbyte's.

El peso de los datos GPS enviados puede variar por la transformación binaria en su respectivo protocolo de comunicación, y la encriptación que utiliza GCM para poder transportarlos.

5.6.2.4. Evaluación del consumo datos en el envío de solicitudes y consultas para la recuperación de usuarios por parte del Dispositivo Móvil.

- **Descripción:** En el presente estudio se recoge información de la carga y descarga de datos que utiliza el dispositivo móvil para consultar los usuarios registrados por el administrador de la aplicación en el servidor de Base de Datos.

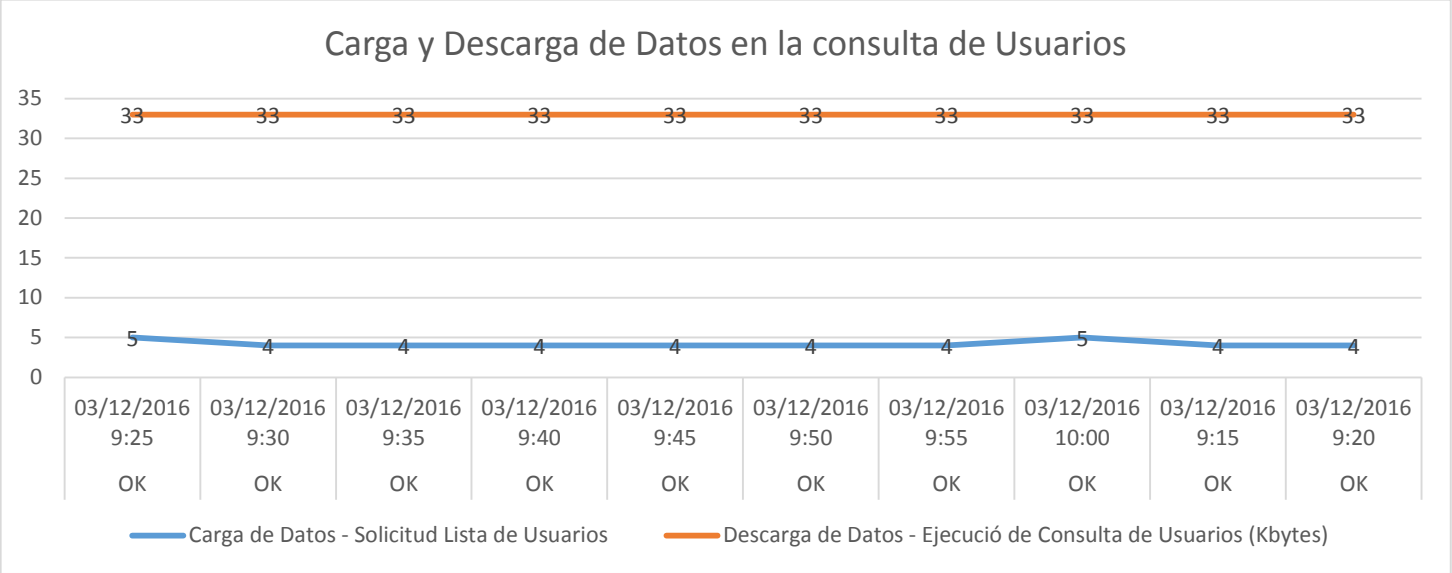


Ilustración V-80: Consumo de Datos al consultar usuarios registrados por parte del Dispositivo Móvil

Elaborado por: Diego Ponce – Juan Prado

- **Notas:** En base a los datos obtenidos de la tabla 46 de Anexos, se analizan 10 transacciones que refleja el consumo de datos (Kbyte's) para que el dispositivo móvil logré consultar los usuarios registrados en el servidor de Base de Datos.

Los datos fueron tomados cada 5 minutos dando los siguientes resultados, tomando en cuenta que el dispositivo uso una conexión WI-FI:

- El promedio de solicitudes para consulta de usuarios fue de 4,2 Kbyte's en el lapso de 1 hora.
- La solicitud que tuvo menor consumo fue de 4 Kbyte's.
- La solicitud que tuvo mayor consumo fue de 5 Kbyte's.

- La lista de usuarios con un total de 7 personas registradas ha consumido 33 Kbyte'es fijos en todas las transacciones.

La longitud de datos en las solicitudes de usuarios puede variar, ya que algunas activan el servicio web en el servidor de aplicaciones, y en otras ocasiones puede haber reenvío de datos por desconexión de comunicación o falla de servicios por parte del servidor.

5.6.2.5. Evaluación del consumo datos en la descarga de mapas digitales en la Aplicación Móvil.

- **Descripción:** En el presente estudio se reúne información sobre los datos consumido en la carga de solicitudes para descarga de mapas desde la nube y descarga de las mismas.

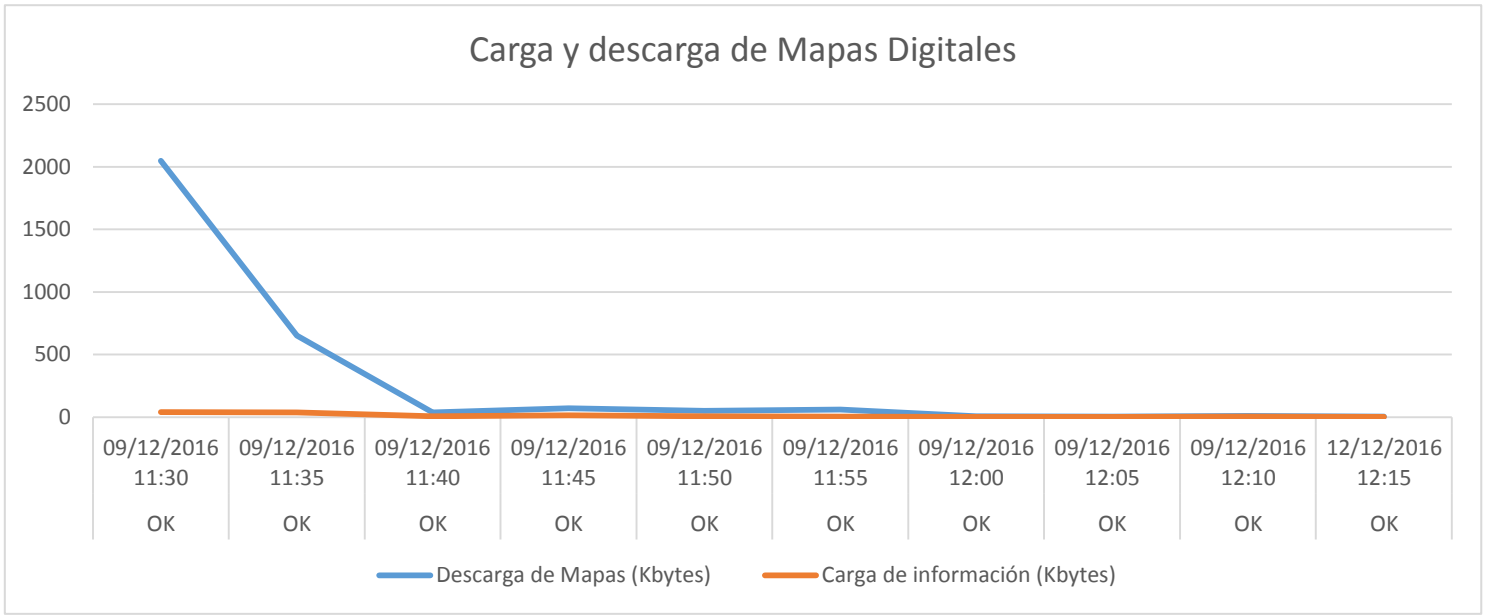


Ilustración V-81: Consumo de información en la descarga de mapas digitales por parte del Dispositivo Móvil

- **Notas:** En base a los datos obtenidos de la tabla 47 de Anexos, se analizan 10 transacciones que refleja el consumo de datos (Kbyte's) para que el dispositivo móvil logré visualizar los mapas digitales en la aplicación.

Los datos fueron tomados cada 5 minutos dando los siguientes resultados, tomando en cuenta que el dispositivo uso una conexión WI-FI:

- El promedio de solicitudes de mapas digitales fue de 11,5 Kbyte's en el lapso de 1 hora.
- La solicitud que tuvo menor consumo fue de 1 Kbyte's.
- La solicitud que tuvo mayor consumo fue de 41 Kbyte's.

- El promedio de consumo para la descarga de mapas digitales como tal fue de 294,1 Kbyte's.
- El valor mínimo consumido fue de 4 Kbyte's.
- El valor máximo consumido fue de 2048 KBytes's.

Cabe mencionar que el valor máximo que como se muestra en la tabla 47 de Anexos, el valor máximo de descarga de un mapa digital se manifiesta solo en su primera solicitud y demuestra que, con 2048 Kbyte´s se puede descargar un área aproximada de 2900 Km.^{2} visualizado a una altura aproximada de 500 m. en el modo de visualización predeterminado que dispone el API de Google Maps.

De igual manera se aprecia en caso adicional, que después de hacer una primera descarga, las siguientes serán de menor peso o longitud como se demuestra que con 650 Kbyte's se lográ descargar 2900 Km. cuadrados de un mapa digital en una altura de 500 m. por el motivo que la herramienta API solamente está construyendo complementos extras como calles, avenidas y lugares nombrados.

5.1.1.1. Evaluación del consumo batería en el uso de la aplicación móvil junto al Sistema Operativo Android.

- **Descripción:** El presente estudio tiene datos para la evaluación del consumo de batería en el dispositivo móvil tomando en cuenta que solo se ejecuta el Sistema Operativo Android y la aplicación móvil con sus respectivos servicios y procesos en segundo plano, como dato adicional se menciona que el dispositivo usado para las pruebas fue un Samsung Galaxy S4 mini, el cual tiene una batería de 1900 mAh.

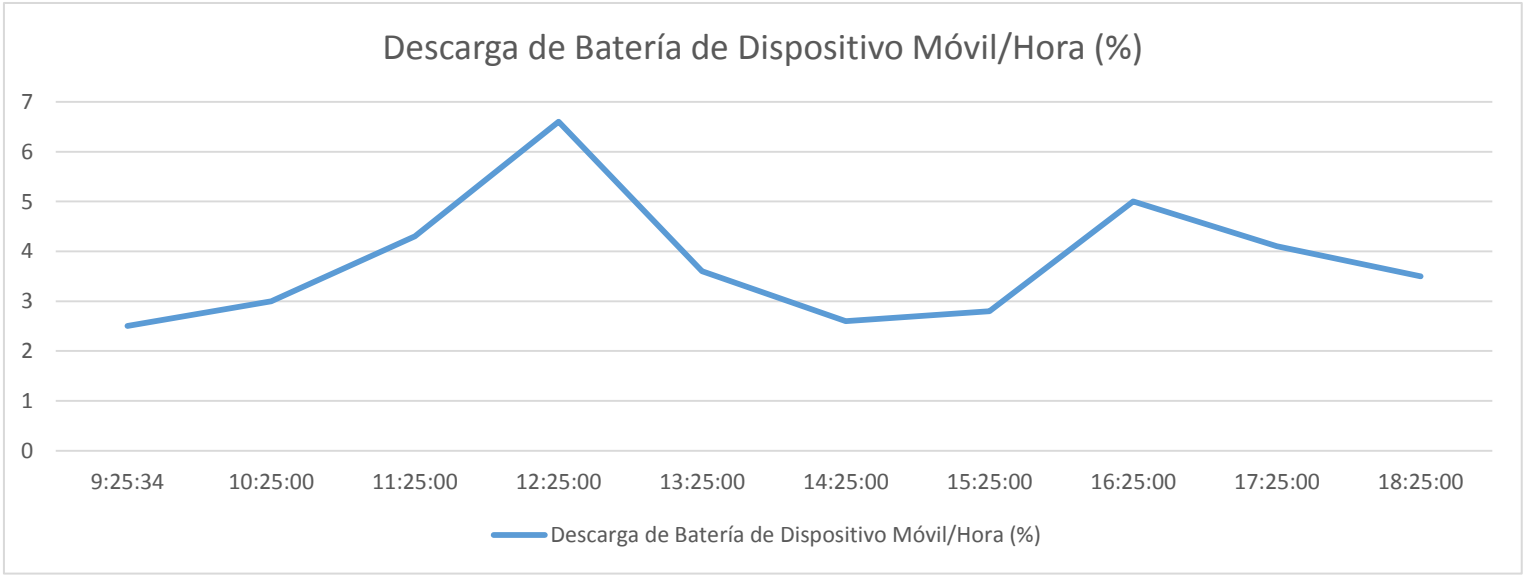


Ilustración V-82: Consumo de batería por parte del Dispositivo Móvil

- **Notas:** En base a los datos obtenidos de la tabla 48 de Anexos, se analizan 10 transacciones que refleja el consumo de datos (Kbyte's) para que el dispositivo móvil logré visualizar los mapas digitales en la aplicación.

Los datos fueron tomados cada 60 minutos dando los siguientes resultados.

- La aplicación móvil conjunto al sistema operativo Android ha consumido un total de 38% de batería en lapso de 10 hrs.
- El consumo mayor por hora ha sido de 6,6%.
- El consumo menor por hora ha sido de 2,5%.
- En promedio a las 10 horas estudiadas la aplicación móvil consume 3,8% de energía en una batería de 1900 mAh.

5.7. ESTUDIO DE RESULTADOS

Una vez recolectado toda la información en el estudio de pruebas, se puede considerar y definir varias conclusiones que proponen visión y factibilidad para posibles implantaciones del proyecto de software hacia entornos de trabajo reales. Este estudio será clasificado en las siguientes categorías.

5.7.1. Plataforma Web

Se toma en cuenta una conexión mínima de 100 usuarios en paralelo y una administración de 1000 conexiones diarias, para esto se considera los siguientes requerimientos.

5.7.1.1. Requerimientos mínimos de Hardware

Para la implantación de servidores tanto de aplicaciones como base de datos se tiene en conideración el equipo utilizado para el desarrollo y pruebas del proyecto de software en

cuanto a la plataforma web, adicional, toda la información recolectada en el apartado de pruebas.

- **Servidor de Aplicaciones y Base de Datos**
  - Procesador dual de 2.5 Ghz de velocidad.
  - 1 MB de Cache, Bus 1333 Mhz.
  - 4 GB de RAM.
  - 250 GB Disco duro (2 RAID mínimo).
  - Acceso a internet vía Ethernet.

5.7.1.2. Requerimientos mínimos de Software

El entorno de pruebas está optimizado para trabajar en varios tipos de sistema operativo.

- **Servidor de Aplicaciones**
  - Windows Server, versión 2012 o superior.
  - Linux OS, derivados Centos, Debian, Gentoo, Ubuntu Server, Fedora, etc.
  - Motor de Servidor de Aplicaciones JBoss 7 SA o superior.
  - Navegador Google Chrome versión 53 o superior.
  - Navegador Mozilla Firefox versión 21 o superior.
  - Java JDK y JRE versión 8 o superior.

- **Servidor de Base de Datos**

  - Windows Server, versión 2012 o superior.
  - Linux OS, derivados Centos, Debian, Gentoo, Ubuntu Server, Fedora, etc.
  - Motor de base de datos MySQL 4 o superior.
  - Navegador Google Chrome versión 53 o superior.
  - Navegador Mozilla Firefox versión 21 o superior.

5.7.2. Requerimientos de Comunicación y Acceso a Internet

Una vez hecha la recolección de información y evaluaciones en el apartado de Pruebas se tiene como un factor crítico la estabilidad de la conexión de internet y su manejo en su tráfico de datos, por tanto, se define los siguientes parámetros para contratación de servicios a una ISP determinada.

- Compartición corporativa: 1:1 o doméstica 8:1.
- Velocidad de Descarga: 5 mbytes/seg.
- Velocidad de Carga: 5 Mbytes/seg.
- Arquitectura LAN alámbrica.

5.7.3. Aplicación Móvil

5.7.3.1. Requerimientos mínimos de Hardware

- Procesador ARM a 1 Ghz.
- 1 GB. De memoria RAM.
- 8 GB de memoria de Almacenamiento.
- GPS con soporte A-GPS, GLONASS.
- Conexión a datos Wi-Fi, Conectividad WPA, 3G, o LTE

- Batería de 1900 mAh.

5.7.3.2. Requerimientos mínimos de Software

- Android versión 4.x o mayor.
- Servicios de Google Play actualizado a la fecha, mínimo API 21
- Acceso a Internet

5.7.3.3. Requerimientos sobre el Acceso a internet

Tomando como referencia el estudio de pruebas de la aplicación móvil se comprueba que la misma usa alrededor 24 Kbytes en una hora y un consumo mensual de 5,8 Mbytes que representa apenas el 0,5% de un plan promedio básico de telefonía celular en Ecuador que corresponde a 1353 Mbytes con un costo de 20 usd. mensuales

CAPÍTULO VI.

CONCLUSIONES Y RECOMENDACIONES

6.1. CONCLUSIONES

- Las redes inalámbricas WI-FI cumplen con varias funciones críticas al momento de la compartición de recursos entre equipos y áreas de comunicación, el acceso a internet que puede brindar en ellos y también en tareas complementarias como la ayuda al proceso de ubicación GPS como se ha apreciado en el transcurso de

este trabajo de disertación. Estas herramientas pueden ser aprovechadas y aplicadas en el desarrollo de aplicaciones para monitorización de dispositivos clientes que cuenten con tecnología de posicionamiento global y administrados remotamente.

- Para el proceso de ubicación GPS vía WI-FI, una de las herramientas imprescindibles es al menos 3 routers cercanos que ayuden a ejecutar la triangularzación del dispositivo objetivo, con la implementación actual de internet fijo en la mayoría zonas urbanas y rurales, así como comerciales e industriales, es más fácil aprovechar el servicio de ubicación por este medio, reduciendo así, costos por consumo de datos móviles y siendo más rentables para empresas y organizaciones.

- En la actualidad existen varias herramientas de libre uso para desarrolladores que pueden ser explotadas para la creación e implementación de aplicaciones empresariales, tales son los ejemplos vistos como las API's de Google Map's (GMAPS), que ayuda al proceso de geo posicionamiento y trae consigo el complemento de mapas digitales lo que incrementa la usabilidad de las aplicaciones web y móviles, así como Google Cloud Messaging (GCM) que ha colaborado en el envió de mensajes y ubicaciones para la gestión y monitoreo de los dispositivos.

- El prototipo de la aplicación para la ubicación y monitoreo de dispositivos móviles tiene un gran margen de mercado y escalabilidad de implementación empresarial, pues tanto su código y arquitectura tiene la posibilidad de acoplarse a cualquier lógica de negocio que puede requerirse a un nivel corporativo.

- Los requerimientos hardware y software para empresa u organización serán diferentes y tendrán que ser implementadas bajo sus propias necesidades y lógica de negocio, la implantación del proyecto de software no tendrá el mismo proceso y equipo en un banco o entidad financiera, que, en una papelería o mini centro comercial, también dependerá de esto las políticas sobre el uso y manejo de la información en cada una de ellas.

6.2. RECOMENDACIONES

- El servicio de GPS vía WIF en el caso de Google Maps, es un servicio relativamente gratis, ya que utiliza routers en conjunto sin necesidad de tener acceso a ellos, y es de gran utilidad en la necesidad de usuarios finales como de aplicaciones gestoras del chip de posicionamiento, por esto, es recomendable para los municipios de zonas urbanas, rurales, comerciales e industriales, la implementación de puntos de acceso a internet gratuitos o públicos, ya sea en estaciones de transporte, parques, terminales de transporte, centros comerciales, así como en zonas ecológicas y apartadas para que este servicio sea más efectivo, rápido y preciso.

- De igual manera sería recomendable que las organizaciones dedicadas a la cartografía y digitalización de mapas digitales, principalmente Google a enfocar sus procesos y trabajos en zonas alejadas de las ciudades, menos pobladas y de difícil acceso, pues la demanda de sus servicios y el mercado de aplicaciones en aquellas zonas es reducido pero no abandonado, por aquello muchas veces es difícil implantar

software móvil y de acceso remoto en algunas zonas por falta de servicios SIG o acceso a internet.

- Antes de realizar una implantación del proyecto de software a una empresa o entidad, es recomendable realizar una evaluación previa y un análisis completo del negocio de la misma, con la finalidad de determinar el equipo de hardware adecuado para que las aplicaciones funcionen de manera adecuada, estable y eficiente sin importar el número de conexiones y requerimientos de red que llegue a tener.

CAPÍTULO VII.

ANEXOS

Tabla VII-1: Datos de Monitoreo: Tiempo de respuesta del Servidor de Aplicaciones y su respectivo cliente (10 Horas)

Estado	Fecha	Tiempo de Respuesta (ms)	Código de Respuesta (Query)	Bytes Transferidos
OK	01/11/2016 10:02	6	200	2432
OK	01/11/2016 9:57	6	200	2432
OK	01/11/2016 9:52	6	200	2432
OK	01/11/2016 9:47	7	200	2432
OK	01/11/2016 9:42	6	200	2432

OK	01/11/2016 9:37	8	200	2432
OK	01/11/2016 9:32	5	200	2432
OK	01/11/2016 9:27	5	200	2432
OK	01/11/2016 9:23	6	200	2432
OK	01/11/2016 9:18	7	200	2432
OK	01/11/2016 9:13	7	200	2432
OK	01/11/2016 9:08	18	200	2432
OK	01/11/2016 9:02	8	200	2432
OK	01/11/2016 8:57	7	200	2432
OK	01/11/2016 8:52	5	200	2432
OK	01/11/2016 8:47	16	200	2432
OK	01/11/2016 8:42	7	200	2432
OK	01/11/2016 8:37	5	200	2432
OK	01/11/2016 8:32	8	200	2432
OK	01/11/2016 8:27	15	200	2432
OK	01/11/2016 8:22	7	200	2432
OK	01/11/2016 8:17	6	200	2432
OK	01/11/2016 8:12	6	200	2432
OK	01/11/2016 8:07	6	200	2432
OK	01/11/2016 8:02	7	200	2432
OK	01/11/2016 7:57	6	200	2432
OK	01/11/2016 7:52	5	200	2432
OK	01/11/2016 7:47	7	200	2432
OK	01/11/2016 7:42	6	200	2432
OK	01/11/2016 7:37	5	200	2432
OK	01/11/2016 7:32	7	200	2432
OK	01/11/2016 7:27	6	200	2432
OK	01/11/2016 7:22	7	200	2432
OK	01/11/2016 7:17	12	200	2432
OK	01/11/2016 7:12	7	200	2432
OK	01/11/2016 7:07	6	200	2432
OK	01/11/2016 7:02	5	200	2432
OK	01/11/2016 6:57	6	200	2432
OK	01/11/2016 6:52	6	200	2432
OK	01/11/2016 6:47	5	200	2432
OK	01/11/2016 6:42	121	200	2432

OK	01/11/2016 6:37	6	200	2432
OK	01/11/2016 6:32	6	200	2432
OK	01/11/2016 6:27	7	200	2432
OK	01/11/2016 6:22	7	200	2432
OK	01/11/2016 6:17	6	200	2432
OK	01/11/2016 6:12	91	200	2432
OK	01/11/2016 6:07	4	200	2432
OK	01/11/2016 6:02	6	200	2432
OK	01/11/2016 5:57	6	200	2432
OK	01/11/2016 5:52	7	200	2432
OK	01/11/2016 5:47	4	200	2432
OK	01/11/2016 5:42	8	200	2432
OK	01/11/2016 5:37	6	200	2432
OK	01/11/2016 5:32	8	200	2432
OK	01/11/2016 5:27	6	200	2432
OK	01/11/2016 5:22	7	200	2432
OK	01/11/2016 5:17	24	200	2432
OK	01/11/2016 5:12	7	200	2432
OK	01/11/2016 5:07	6	200	2432
OK	01/11/2016 5:02	5	200	2432
OK	01/11/2016 4:57	6	200	2432
OK	01/11/2016 4:52	6	200	2432
OK	01/11/2016 4:47	7	200	2432
OK	01/11/2016 4:42	5	200	2432
OK	01/11/2016 4:37	6	200	2432
OK	01/11/2016 4:32	5	200	2432
OK	01/11/2016 4:27	7	200	2432
OK	01/11/2016 4:22	16	200	2432
OK	01/11/2016 4:17	4	200	2432
OK	01/11/2016 4:12	6	200	2432
OK	01/11/2016 4:07	5	200	2432
OK	01/11/2016 4:02	7	200	2432
OK	01/11/2016 3:57	5	200	2432
OK	01/11/2016 3:52	6	200	2432
OK	01/11/2016 3:47	4	200	2432
OK	01/11/2016 3:42	6	200	2432

OK	01/11/2016 3:37	6	200	2432
OK	01/11/2016 3:32	6	200	2432
OK	01/11/2016 3:27	5	200	2432
OK	01/11/2016 3:22	7	200	2432
OK	01/11/2016 3:17	6	200	2432
OK	01/11/2016 3:12	5	200	2432
OK	01/11/2016 3:07	6	200	2432
OK	01/11/2016 3:02	7	200	2432
OK	01/11/2016 2:57	6	200	2432
OK	01/11/2016 2:52	7	200	2432
OK	01/11/2016 2:47	7	200	2432
OK	01/11/2016 2:42	6	200	2432
OK	01/11/2016 2:37	6	200	2432
OK	01/11/2016 2:32	7	200	2432
OK	01/11/2016 2:27	4	200	2432
OK	01/11/2016 2:22	8	200	2432
OK	01/11/2016 2:17	7	200	2432
OK	01/11/2016 2:12	6	200	2432
OK	01/11/2016 2:07	7	200	2432
OK	01/11/2016 2:02	13	200	2432
OK	01/11/2016 1:57	4	200	2432
OK	01/11/2016 1:52	6	200	2432
OK	01/11/2016 1:47	7	200	2432
OK	01/11/2016 1:42	5	200	2432
OK	01/11/2016 1:37	7	200	2432
OK	01/11/2016 1:32	6	200	2432
OK	01/11/2016 1:27	6	200	2432
OK	01/11/2016 1:22	6	200	2432
OK	01/11/2016 1:17	13	200	2432
OK	01/11/2016 1:12	27	200	2432
OK	01/11/2016 1:07	6	200	2432
OK	01/11/2016 1:02	6	200	2432
OK	01/11/2016 0:57	5	200	2432
OK	01/11/2016 0:52	6	200	2432
OK	01/11/2016 0:47	6	200	2432
OK	01/11/2016 0:42	5	200	2432

OK	01/11/2016 0:37	6	200	2432
OK	01/11/2016 0:32	6	200	2432
OK	01/11/2016 0:27	6	200	2432
OK	01/11/2016 0:22	4	200	2432
OK	01/11/2016 0:17	4	200	2432
OK	01/11/2016 0:12	6	200	2432
OK	01/11/2016 0:07	6	200	2432
OK	01/11/2016 0:02	5	200	2432

Total Transacciones Estudiadas	**121**
Prom. Tiempo de Respuesta (ms)	**9,45**
Mín. Tiempo de Respuesta (ms)	**4**
Max. Tiempo de Respuesta (ms)	**121**

Elaborado por: Diego Ponce – Juan Prado

Tabla VII-2: Datos de Monitoreo: Tiempo de respuesta del Servidor de Aplicaciones y su respectivo cliente (14 Días)

Estado de Transacción	Tiempo de Actividad (%)	Fecha	Tiempo de Respuesta (ms)	Longitud (Bytes)
OK	100	2016 dic. 11	4,14	2432
OK	100	2016 dic. 10	4,12	2432
OK	100	2016 dic. 09	4,33	2432
OK	100	2016 dic. 08	4,85	2432
OK	100	2016 dic. 07	4,89	2432
OK	100	2016 dic. 06	4,07	2432
OK	100	2016 dic. 05	4,25	2432
OK	100	2016 dic. 04	4,08	2432

OK	100	2016 dic. 03	4,1	2432
OK	100	2016 dic. 02	4,46	2432
OK	90,9	2016 dic. 01	4,11	2210,9
OK	100	2016 nov. 30	157,13	2432
OK	100	2016 nov. 29	21,9	2432
OK	100	2016 nov. 28	12,36	2432
Promedio	99,35	**Promedio**	17,06	2416,21
		Valor Máximo	157,13	2432,00
		Valor Mínimo	4,07	2210,90

Elaborado por: Diego Ponce – Juan Prado

Tabla VII-3: Monitoreo de ancho de banda consumido por el Servidor de Aplicaciones (10 Horas)

Estado	Fecha	Trafico de Salida (Bytes/Seg.)	Tráfico de Entrada (Bytes/Seg.)
OK	01/11/2016 10:22	22481,69	2773,25
OK	01/11/2016 10:17	5464,74	2235,91
OK	01/11/2016 10:12	5133,72	2369,87
OK	01/11/2016 10:07	4721,86	2151,57
OK	01/11/2016 10:02	4756,01	2146,11
OK	01/11/2016 9:57	4756,2	2103,16
OK	01/11/2016 9:52	4918,06	2096,29

OK	01/11/2016 9:47	4714,5	2052,92
OK	01/11/2016 9:42	5115,77	2302,29
OK	01/11/2016 9:37	16502,13	2543,94
OK	01/11/2016 9:32	5510,28	2103,45
OK	01/11/2016 9:27	5146,95	2124,59
OK	01/11/2016 9:22	6336,58	2218,77
OK	01/11/2016 9:17	6430,48	2399,4
OK	01/11/2016 9:12	4711,61	2388,02
OK	01/11/2016 9:07	5760,4	3448,82
OK	01/11/2016 9:02	5515,09	2293,31
OK	01/11/2016 8:57	4245,52	2016,72
OK	01/11/2016 8:52	3638,87	2051,04
OK	01/11/2016 8:47	3136,14	1907,13
OK	01/11/2016 8:42	3696,76	2042,83
OK	01/11/2016 8:37	3753,83	2149,34
OK	01/11/2016 8:32	3581,87	1886,02
OK	01/11/2016 8:27	3550,95	1893,98
OK	01/11/2016 8:22	3724,04	1914,8
OK	01/11/2016 8:17	4290,82	2071,51
OK	01/11/2016 8:12	4102,71	1983,03
OK	01/11/2016 8:07	4373,11	2221,96
OK	01/11/2016 8:02	16851,08	4389,24
OK	01/11/2016 7:57	3756,73	1895,06
OK	01/11/2016 7:52	3951,14	1988,11
OK	01/11/2016 7:47	4028,38	1959,52
OK	01/11/2016 7:42	3686,56	1898,64
OK	01/11/2016 7:37	3901,01	2065,96
OK	01/11/2016 7:32	3871,33	1893,01
OK	01/11/2016 7:27	4038,79	1937,83
OK	01/11/2016 7:22	4362,83	1919,79
OK	01/11/2016 7:17	4443,4	1988,91
OK	01/11/2016 7:12	5287,17	2081,47
OK	01/11/2016 7:07	3127,32	2075,61
OK	01/11/2016 7:02	2973,8	1905,73
OK	01/11/2016 6:57	2886,39	1896,45
OK	01/11/2016 6:52	3366,54	2027,56

OK	01/11/2016 6:47	3121,99	1995,89
OK	01/11/2016 6:42	3120,77	1939,9
OK	01/11/2016 6:37	3126,56	2067,17
OK	01/11/2016 6:32	2921,65	1930,32
OK	01/11/2016 6:27	3037,76	1968,84
OK	01/11/2016 6:22	3134,46	1966,11
OK	01/11/2016 6:17	2972,33	1958,37
OK	01/11/2016 6:12	2993,65	2058,61
OK	01/11/2016 6:07	3007,08	1943,97
OK	01/11/2016 6:02	3023,03	1954,65
OK	01/11/2016 5:57	2939,6	1910,24
OK	01/11/2016 5:52	3084,88	1937,63
OK	01/11/2016 5:47	2834,28	1871,56
OK	01/11/2016 5:42	3090,09	2019,76
OK	01/11/2016 5:37	23810,13	2513,97
OK	01/11/2016 5:32	2957,66	1890,14
OK	01/11/2016 5:27	2955,2	1938,3
OK	01/11/2016 5:22	3127,4	1942,04
OK	01/11/2016 5:17	2915,93	1932,74
OK	01/11/2016 5:12	3164,08	2004,6
OK	01/11/2016 5:07	3153,74	2082,26
OK	01/11/2016 5:02	2930,12	1883,41
OK	01/11/2016 4:57	2860,99	1900,09
OK	01/11/2016 4:52	3151,63	1994,57
OK	01/11/2016 4:47	3193,46	2016,83
OK	01/11/2016 4:42	3064,07	1985,3
OK	01/11/2016 4:37	3070,82	2063,36
OK	01/11/2016 4:32	2879,3	1873,74
OK	01/11/2016 4:27	2829,24	1850,37
OK	01/11/2016 4:22	3212,39	1988,57
OK	01/11/2016 4:17	2959,87	1916,13
OK	01/11/2016 4:12	3382,26	2143,93
OK	01/11/2016 4:07	2964,02	1987,92
OK	01/11/2016 4:02	3035,3	1949,6
OK	01/11/2016 3:57	3149,9	1969,51
OK	01/11/2016 3:52	3171,43	1981,35

OK	01/11/2016 3:47	2908,84	1886,09
OK	01/11/2016 3:42	2990,32	1988,53
OK	01/11/2016 3:37	2982,93	1974,99
OK	01/11/2016 3:32	3158,27	1943,31
OK	01/11/2016 3:27	2898,67	1881,51
OK	01/11/2016 3:22	2981,55	1929,52
OK	01/11/2016 3:17	3001,39	1894,47
OK	01/11/2016 3:12	3421,7	2179,27
OK	01/11/2016 3:07	3132,18	2030
OK	01/11/2016 3:02	2862,43	1886,45
OK	01/11/2016 2:57	2903,13	1906,92
OK	01/11/2016 2:52	3059,77	1931,99
OK	01/11/2016 2:47	2949,23	1901,03
OK	01/11/2016 2:42	2938,24	1976,8
OK	01/11/2016 2:37	2837,17	1817,37
OK	01/11/2016 2:32	2865,33	1848,9
OK	01/11/2016 2:27	2918,49	1906,38
OK	01/11/2016 2:22	3099,22	1992,73
OK	01/11/2016 2:17	3199,11	2008,67
OK	01/11/2016 2:12	3402,63	2179,84
OK	01/11/2016 2:07	3006,66	1930,33
OK	01/11/2016 2:02	2955,73	1936,73
OK	01/11/2016 1:57	2908,13	1893,02
OK	01/11/2016 1:52	3059,7	1925,39
OK	01/11/2016 1:47	3050,43	1898,5
OK	01/11/2016 1:42	2996,76	1991,63
OK	01/11/2016 1:37	23655,63	2398,74
OK	01/11/2016 1:32	2930,67	1938,34
OK	01/11/2016 1:27	3042,49	1965,5
OK	01/11/2016 1:22	3013,26	1938,45
OK	01/11/2016 1:17	3090,92	1943,25
OK	01/11/2016 1:12	3362,96	2113,57
OK	01/11/2016 1:07	2937,14	1930,7
OK	01/11/2016 1:02	2926,42	1932,47
OK	01/11/2016 0:57	2944,18	1956,35
OK	01/11/2016 0:52	3002,12	1875,51

OK	01/11/2016 0:47	2976,39	1912,42
OK	01/11/2016 0:42	2944,57	1994,47
OK	01/11/2016 0:37	3167,29	2016,95
OK	01/11/2016 0:32	2902,58	1895,62
OK	01/11/2016 0:27	2886,85	1882,58
OK	01/11/2016 0:22	2987,12	1935,33
OK	01/11/2016 0:17	3003,02	1915,6
OK	01/11/2016 0:12	3297,98	2146,35
OK	01/11/2016 0:07	2970,48	1978,01
OK	01/11/2016 0:02	2936,98	1890,34
	Total de Transferencias	**125**	
	Prom. Transferencia	**4163,34**	**2042,76**
	Resultado Mínimo	**2829,24**	**1817,37**
	Resultado Máximo	**23810,13**	**4389,24**

Elaborado por: Diego Ponce – Juan Prado

Tabla VII-4: Monitoreo de ancho de banda consumido por el Servidor de Aplicaciones (14 Días)

Estado	Fecha	Trafico de Salida (Bytes/Seg)	Tráfico de Entrada (Bytes/Seg)
OK	11/dic./2016	3522,88	6979,31
OK	10/dic./2016	3605,98	7137,12
OK	09/dic./2016	3760,92	10997,25
OK	08/dic./2016	3618,06	8221,97
OK	07/dic./2016	3602,53	8101,29
OK	06/dic./2016	3558,9	7457,69
OK	05/dic./2016	3665,97	7613,13

OK	04/dic./2016	3529,71	7157,74
OK	03/dic./2016	3585,46	7152,07
OK	02/dic./2016	3771,89	8900,7
OK	01/dic./2016	3836,38	10071,27
OK	30/nov./2016	2228,17	5508,83
OK	29/nov./2016	2274,76	5523,41
OK	28/nov./2016	2430,28	5906,85
	Total de Transferencias	14	
	Promedio Transferencias	7623,47	3356,56
	Resultado Mínimo	5508,83	2228,17
	Resultado Máximo	10997,25	3836,38

Elaborado por: Diego Ponce – Juan Prado

Tabla VII-5: Monitoreo de Memoria RAM del Servidor de Aplicaciones

Estado	Fecha	Memoria Libre (MB)	Memoria Usada (MB)	Memoria Física Total (MB)	Memoria Virtual Libre (MB)	Total Memoria (MB)
OK	11/dic./2016	1059,58	2941,32	4000,92	3607,44	7840,89
OK	10/dic./2016	1119,81	2881,09	4000,91	3618,57	7840,9
OK	09/dic./2016	1185,87	2815,03	4000,91	3579,38	7840,9
OK	08/dic./2016	1277,16	2723,74	4000,91	3607,22	7840,9
OK	07/dic./2016	1367,3	2633,6	4000,91	3689,86	7840,9
OK	06/dic./2016	1384,62	2616,28	4000,91	3739,48	7840,9

OK	05/dic./2016	1371,33	2629,58	4000,91	3767,33	7840,9
OK	04/dic./2016	951,15	3049,76	4000,91	3787,99	7840,9
OK	03/dic./2016	978,81	3022,09	4000,91	3822,19	7840,9
OK	02/dic./2016	1057,31	2943,59	4000,91	3799,01	7840,9
OK	01/dic./2016	1110,85	2890,06	4000,9	3672,91	7840,92
OK	30/nov./2016	406,94	3593,97	4000,9	1463,55	8284,3
OK	29/nov./2016	435,21	3565,7	4000,91	1505,28	8286,94
	Total	13,00				
	Promedio	1054,30	2946,60	4000,91	3358,48	7909,32
	Resultado Mín.	406,94	2616,28	4000,90	1463,55	7840,89
	Resultado Máx.	1384,62	3593,97	4000,92	3822,19	8286,94

Elaborado por: Diego Ponce – Juan Prado

Tabla VII-6: Monitoreo de Carga de Datos en el envío de Mensajes de la Aplicación Móvil

Estado	Fecha	Carga de Datos - Envío de Mensajes (Kbytes)
OK	01/12/2016 9:25	1
OK	01/12/2016 9:30	2
OK	01/12/2016 9:35	2
OK	01/12/2016 9:40	2
OK	01/12/2016 9:45	6
OK	01/12/2016 9:50	2

OK	01/12/2016 9:55	1
OK	01/12/2016 10:00	2
OK	01/12/2016 9:15	2
OK	01/12/2016 9:20	2
	Total	10
	Promedio	2,2
	Resultado Mín.	1
	Resultado Máx.	6

Elaborado por: Diego Ponce – Juan Prado

Tabla VII-7: Monitoreo de Descarga de Datos en la recepción de mensajes del Dispositivo Móvil

Estado	Fecha	Descarga de Datos - Recepción de Mensajes (Kbytes)
OK	01/12/2016 12:15	2
OK	01/12/2016 12:20	4
OK	01/12/2016 12:25	2
OK	01/12/2016 12:30	3
OK	01/12/2016 12:35	2

OK	01/12/2016 12:40	2
OK	01/12/2016 12:45	4
OK	01/12/2016 10:50	2
OK	01/12/2016 12:55	2
OK	01/12/2016 13:00	1
	Total	10
	Promedio	2,4
	Resultado Mín.	1
	Resultado Máx.	4

Elaborado por: Diego Ponce – Juan Prado

Tabla VII-8: Monitoreo de carga de datos en el envío de posición GPS

Estado	Fecha	Carga de Datos - Envío de Posición GPS (Kbytes)
OK	02/12/2026 10:25	4
OK	02/12/2026 10:40	2
OK	02/12/2026 10:45	3
OK	02/12/2026 10:50	2
OK	02/12/2026 10:55	2

OK	02/12/2026 11:00	3
OK	02/12/2026 11:10	2
OK	02/12/2026 11:15	3
OK	02/12/2026 11:20	3
OK	02/12/2026 11:25	2
	Total	10
	Promedio	2,6
	Resultado Mín.	2
	Resultado Máx.	4

Elaborado por: Diego Ponce – Juan Prado

Tabla VII-9: Monitoreo de carga y descarga de datos en la solicitud y consulta respectivamente de usuarios al servidor de Base de Datos

Estado	Fecha	Carga de Datos - Solicitud Lista de Usuarios	Descarga de Datos - Ejecución de Consulta de Usuarios (Kbytes)
OK	03/12/2016 9:25	5	33
OK	03/12/2016 9:30	4	33
OK	03/12/2016 9:35	4	33

OK	03/12/2016 9:40	4	33
OK	03/12/2016 9:45	4	33
OK	03/12/2016 9:50	4	33
OK	03/12/2016 9:55	4	33
OK	03/12/2016 10:00	5	33
OK	03/12/2016 9:15	4	33
OK	03/12/2016 9:20	4	33
	Total	10	
	Promedio	4,2	33
	Resultado Mín.	4	33
	Resultado Máx.	5	33

Elaborado por: Diego Ponce – Juan Prado

Tabla VII-10: Monitoreo de carga y descarga de datos para la visualización de mapas digitales en la aplicación móvil

Estado	Fecha	Descarga de Mapas (Kbytes)	Carga de información (Kbytes)	Area Aprox. (Kilómetros Cuadrádos)	Altura (m)	Modo de Visualización	Observaciones
OK	09/12/2016 11:30	2048	41	2900	500	Predeterminado	1ra. Descarga de Mapa
OK	09/12/2016 11:35	650	38	2900	500	Satelite	1ra. Descarga de Mapa
OK	09/12/2016 11:40	38	6	0,18	50	Predeterminado	

OK	09/12/2016 11:45	70	14	0,18	50	Predeterminado	
OK	09/12/2016 11:50	50	6	0,18	50	Predeterminado	
OK	09/12/2016 11:55	60	4	2900	500	Predeterminado	
OK	09/12/2016 12:00	6	1	1727	200	Predeterminado	
OK	09/12/2016 12:05	4	1	2900	500	Predeterminado	
OK	09/12/2016 12:10	10	3	2900	500	Predeterminado	
OK	12/12/2016 12:15	5	1	0,18	50	Predeterminado	
Total	10						
Promedio	294,1	11,5	1622,772	290			
Resultado Mínimo	4	1	0,18	50			
Resultado Máximo	2048	41	2900	500			

Elaborado por: Diego Ponce – Juan Prado

Tabla VII-11: Monitoreo de Consumo de Batería por parte del Dispositivo Móvil

Estado	Fecha	Descarga de Batería de Dispositivo Móvil/Hora (%)
OK	15/12/2016 9:25	2,5
OK	15/12/2016 10:25	3
OK	15/12/2016 11:25	4,3

OK	15/12/2016 12:25	6,6
OK	15/12/2016 13:25	3,6
OK	15/12/2016 14:25	2,6
OK	15/12/2016 15:25	2,8
OK	15/12/2016 16:25	5
OK	15/12/2016 17:25	4,1
OK	15/12/2016 18:25	3,5
	Total	38
	Promedio	3,8
	Resultado Mín.	2,5
	Resultado Máx.	6,6

Elaborado por: Diego Ponce – Juan Prado

BIBLIOGRAFÍA

Abraham Silberschatz, Herny F. Korth, S. Sudarshan. (2002). Fundamentos de Base de datos. (C. F. Madrid, Ed.) Madrid, España. Obtenido de https://unefazuliasistemas.files.wordpress.com/2011/04/fundamentos-de-bases-de-datos-silberschatz-korth-sudarshan.pdf

Alarcón, V. F. (junio de 2006). Desarrollo de sistemas de Información, una metología basada en el modelado. Cataluña, España: Ediciones UPC. Obtenido de https://books.google.com.ec/books?id=Sqm7jNZS_L0C&pg=PA131&dq=casos+de+uso&hl=es-419&sa=X&ved=0ahUKEwjupt6Ni7rQAhVFLSYKHcmoCiYQ6AEIHjAB#v=onepage&q=casos%20de%20uso&f=false

Antoni Pérez Navarro, Albert Botella, Anna Muñoz Bollas, Rosa Olvella González, Joan Carlos Olmedillas Hernandez, Jesús Rodrigues Lloret. (2011). Introducción a los sistemas de información geográfica y geotelemática. (E. UOC, Ed.) Barcelona, España. Obtenido de https://books.google.com.ec/books?id=xip1wtr8k58C&pg=PA320&lpg=PA320&dq=arquitectura+de+api+google+maps&source=bl&ots=OiigvzIkDo&sig=vy_0a_HuQOU76lrcsfjkHDBGz6w&hl=es&sa=X&ved=0ahUKEwjJqPPYsvrQAhVNyWMKHaB7C_sQ6AEIVjAI#v=onepage&q=arquitectura%20de%20api%2

AUMAILLE, B. (Noviembre de 2002). J2EE, Desarrollo de Aplicaciones Web. (E. ENI, Ed.) Barcelona, España. Obtenido de https://books.google.es/books?hl=es&lr=&id=dsR2ydrU3vUC&oi=fnd&pg=PA5&dq=aplicaciones+web&ots=CtePyCMU5X&sig=eLhQzTPWudjtOLJogLZfooxZmgI#v=onepage&q=aplicaciones%20web&f=false

Barry F. Kavanah, Gelnn Bird J.S. (1989). Surveying: Principles and Applications. New Yersey, USA. Obtenido de http://webdelprofesor.ula.ve/ingenieria/lnova/Archivos/FORMATO-PDF/CAPITULO-10.pdf

Basterra, Bertea, Borello, Castillo, Venturi. (6 de Julio de 2016). Android OS Documentation. Obtenido de https://media.readthedocs.org/pdf/androidos/latest/androidos.pdf

Benbourahala, N. (Julio de 2013). Android 4, Principios del desarrollo de aplicaciones Java. (E. ENI, Ed.) Barcelona, España. Obtenido de https://books.google.es/books?hl=es&lr=&id=JfGIHgcVkfkC&oi=fnd&pg=PA13&dq=sdk+de+android&ots=H9GXJ99ygQ&sig=Z-Mcz0NFtW6URIQDH9-GkXJC6Fc#v=onepage&q=sdk%20de%20android&f=false

Castro, R. (Septiembre de 2005). Avanzando en la seguridad de las redes WI-FI. (B. d. RedIris, Ed.) Obtenido de http://www.rediris.es/difusion/publicaciones/boletin/73/ENFOQUE1.pdf

Fernando Pech-May, Mario A. Gomez-Rodrigez, Luis A de la Cruz-Diaz, Salvador U. Lara-Jerónimo. (2010). Desarrollo de Aplicaciones Web con JPA, EJB, JSF y Prime Faces. (I. T. Rios, Ed.) Villahermosa, Tabasco, México. Obtenido de http://www.tamps.cinvestav.mx/~fpech/sd/files/paper001.pdf

Gilfillan, I. (2003). La Biblia de MySQL. (A. Multimedia, Ed.) Obtenido de https://osmell.files.wordpress.com/2008/08/la-biblia-de-mysql-anaya-multimedia.pdf

González, L. L. (10 de 8 de 2004). El Diseño de Interfaz Gráfica de usuario para publicaciones digitales. (UNAM, Ed.) México. Obtenido de http://www.revista.unam.mx/vol.5/num7/art44/ago_art44.pdf

Ing. Pablo Jara Werchau, Ing. Patricia Nazar. (2015). Estándar IEEE 802.11.X de las WLAN. (E. d. Nacional, Ed.) Tucumán, Argentina. Obtenido de http://www.edutecne.utn.edu.ar/monografias/standard_802_11.pdf

Juan Gabriel Enriquez, Isabel Sandra Casas. (4 de 6 de 2013). Usabilidad en Aplicaciones Móviles. Obtenido de http://secyt.unpa.edu.ar/journal/index.php/ICTUNPA/article/view/ICT-UNPA-62-2013/62

M., L. C. (2002). Topografía Plana. Merida. Obtenido de http://webdelprofesor.ula.ve/ingenieria/lnova/Archivos/FORMATO-PDF/CAPITULO-10.pdf

Mora, S. L. (2002). Programació de aplicaciones web: historia, principios básicos y cientes web. (E. C. Universitario, Ed.) San Vicente, Alicante, España. Obtenido de https://books.google.es/books?hl=es&lr=&id=r9CqDYh2-loC&oi=fnd&pg=PR3&dq=aplicaciones+web&ots=MiDTWm2UC-&sig=dP4_Pf4MTh2qYFi67LcppF-K-dk#v=onepage&q=aplicaciones%20web&f=false

Niola, A. A. (2015). Diseño de Redes Wi-Fi para la ciudad de Guayaquil. (E. S. Litoral, Ed.) Guayaquil, Guayas, Ecuador. Obtenido de https://www.dspace.espol.edu.ec/bitstream/123456789/32035/1/CIUDAD%20DIGITAL%20-%20DISE%C3%91O%20DE%20REDES%20WI-FI%20PARA%20LA%20CIUDAD%20DE%20GUAYAQUIL.pdf

Orero, R. E. (Junio de 2016). Desarrollo de una aplicación móvil para la gestión de itinerancia en dispositivos Android. Santander, España: Unican. Obtenido de https://repositorio.unican.es/xmlui/bitstream/handle/10902/9202/Barcena%20Orero%20Rebeca%20Elena.pdf?sequence=1&isAllowed=y

Pereira, N. E. (Septiembre de 2003). Implementación de un servicio para Geocodificar una Dirección. (F. P. Cataluña, Ed.) Barcelona, España. Obtenido de http://www-cpsv.upc.es/tesines/resumsig_villamizar.pdf

Sanchez, J. (2003). MySQL, Guía Rápida, Versión WIndows. Valladolid, Madrid, España. Obtenido de http://laboratorio.is.escuelaing.edu.co/labinfo/doc/Manual_Basico_de_MySQL.pdf

Sánchez, J. A. (2007). Instalación, Configuración y Administración del Servidor de Aplicaciones JBOSS. Obtenido de https://www.alferez.es/documentos/Jboss.pdf

Saúl Montiel Almeida, J. O. (Diciembre de 2015). Desarrollo de una Aplicación de smartphone para la ubicación de trolebús y aviso de acercamiento. (I. P. Nacional, Ed.) México D.F., México. Obtenido de http://tesis.ipn.mx:8080/xmlui/bitstream/handle/123456789/18552/desarrollo%20de%

20una%20aplicacion%20de%20smartphone%20para%20la%20ubicacion%20de%20trolebus%20y%20aviso%20de%20acercamiento.pdf?sequence=1&isAllowed=y

Sylvain Hébuterne, Sébastien Pérochon. (Octubre de 2014). Android, Guía de Desarrollo de aplicaciones para Smartphones y Tabletas. (E. ENI, Ed.) Barcelona, España. Obtenido de https://books.google.es/books?hl=es&lr=&id=DnQkGjh2H3MC&oi=fnd&pg=PA15&dq=sdk+de+android&ots=G_1ubK3E_Q&sig=Bb9VUI1sxhYlSjwuvkHR5cosxjc#v=onepage&q=sdk%20de%20android&f=false

Universitat Politécnica de Valencia. (2 de 12 de 2010). *Historia de la Informática*. Obtenido de http://histinf.blogs.upv.es/: http://histinf.blogs.upv.es/2010/12/02/historia-de-las-redes-inalambricas/

VB. (s.f.).